

Notes on Ethernet in Saturn

SATURN - DATA
ETSI TRANSPORT SYSTEMS

DRAFT A2

Mick Goodman : JAN 98

Andrew J. Lee Group			
05 FEB 1998			
JP		Index	File

1.0 :INTRODUCTION

2.0 :HISTORY

3.0 :NETWORK SCENARIOS

(For the purpose of seeing if the Datatrib has the right ports, and working out how much differential delay concatenated Vcs must tolerate.)

NOTE: THE FOLLOWING PICS ARE PROVISIONAL, NUMBER AND TYPE OF DATATRIB PORTS HASN'T BEEN FIXED.

- PROVISIONING A DATATRIB AND ENGINEERING IT WITH THE NECESSARY (HTTLESS) FLEXIBILITY IS GOING TO BE A CHALLENGE.
- NEED TO DECIDE IF DATATRIB IS ENGINEERED FROM DAY 1 FOR USE WITH R1 SAT. AGGS OR AGGS WITH STM-4/SLOT.
- THERE IS NO INTENT TO SUPPORT ETHERNET OVER ANY INTERMEDIATE PDH LINKS AS FAR AS THE DATA TRIB IS CONCERNED (LOSE ALL THE BENEFITS OF SDH AND MAJOR JUMP IN COMPLEXITY.)
- THE PHRASE ETHERNET PROTECTION WHICH APPEARS LATER, MEANS I DON'T KNOW WHAT I'M DOING MIGHT BE SPANNING TREE, MIGHT BE SPANNING TREE TRIGGERED BY SDH PATH MONITORING, MIGHT BE SDH CAUSES SWITCHES TO SWAP /DUMP/RELEARN ADDRESS TABLES ETC.

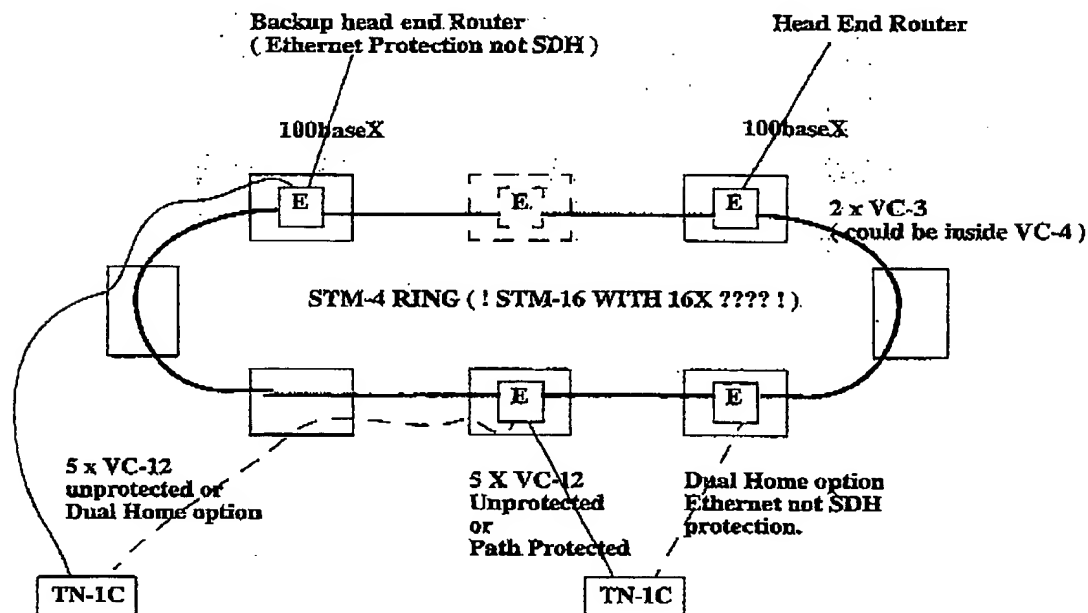
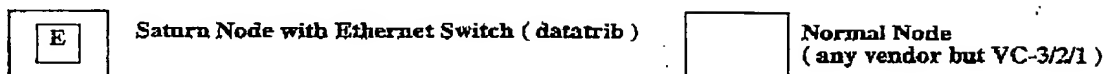
BACKHAUL OPTIONS: VC-4 RING
MATCHED NODE
PHYS TO ROUTER
VC4/3/21/ RING
SATURN 16 WITH EXISTING DATA TRIB
SATURN 16 WITH 100 M ACCESS
MSP SPUR

STM-4 RING (MIXED VENDOR ?)
STM-4 ARC / SPUR ?
4XE ON MSP WITH RING TRIBS

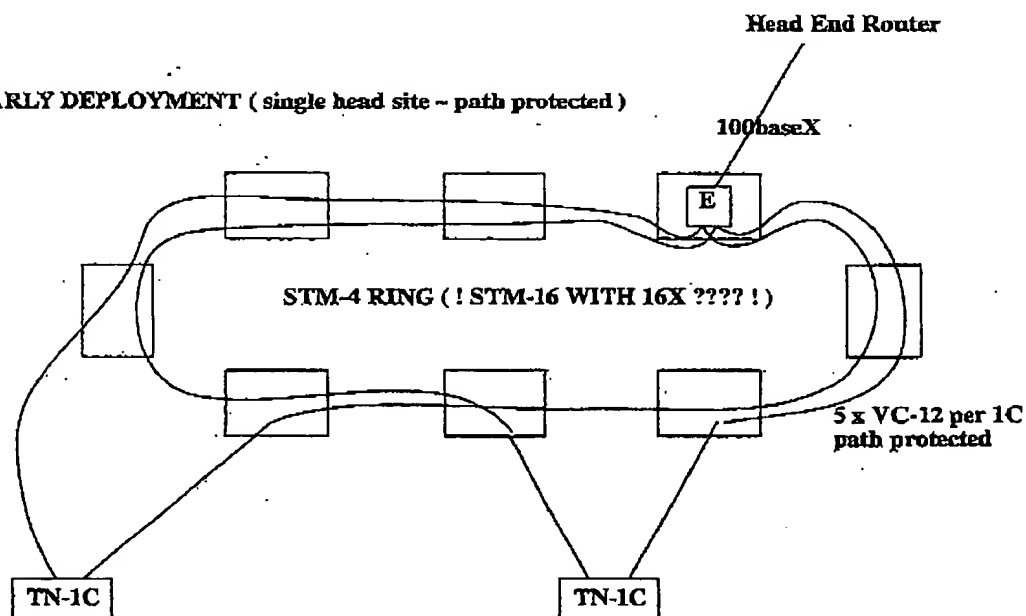
ACCESS:
SPUR TO 1C
1X RING / ARC WITH SPURS TO 1C
1X / 1C RING WITH ACCESS DEVICE
ASCOMS ?

Notes on Ethernet in Saturn

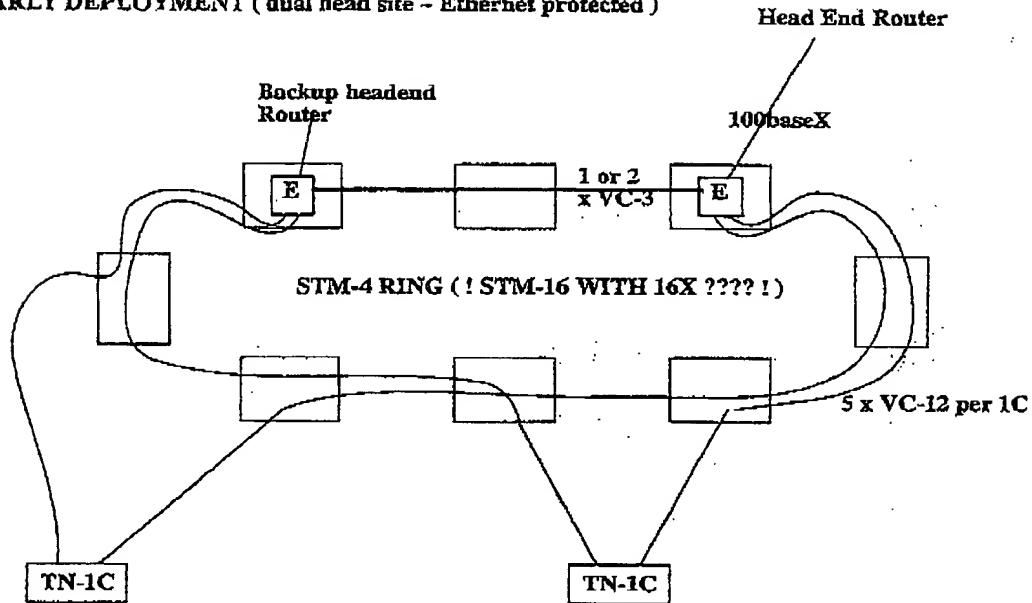
"CONVENTIONAL" CONFIGURATION



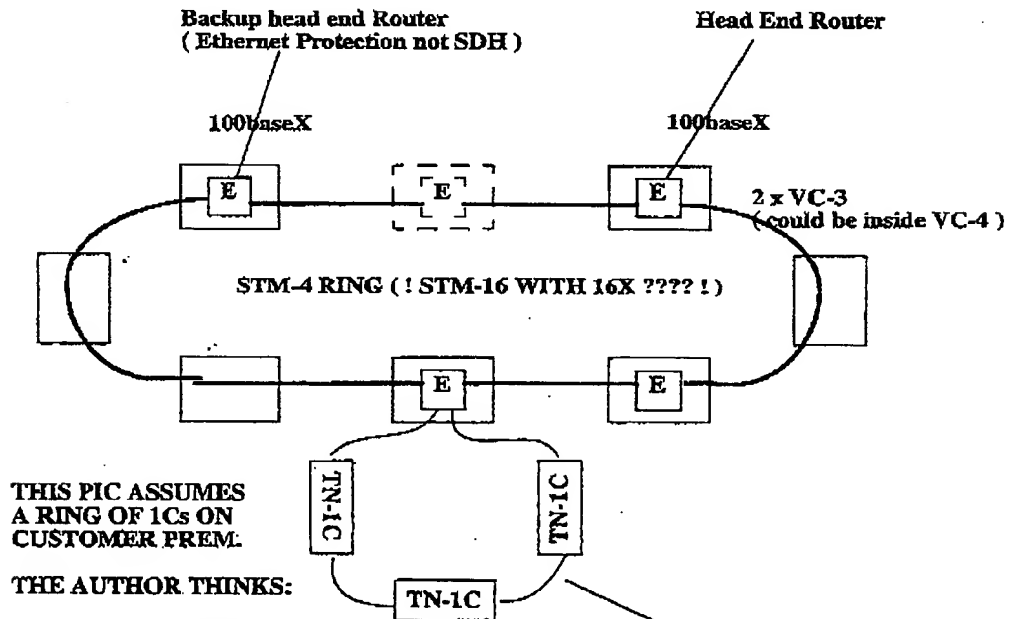
EARLY DEPLOYMENT (single head site - path protected)



EARLY DEPLOYMENT (dual head site - Ethernet protected)



STM-1 RING ACCESS



THIS PIC ASSUMES
A RING OF 1Cs ON
CUSTOMER PREM.

THE AUTHOR THINKS:

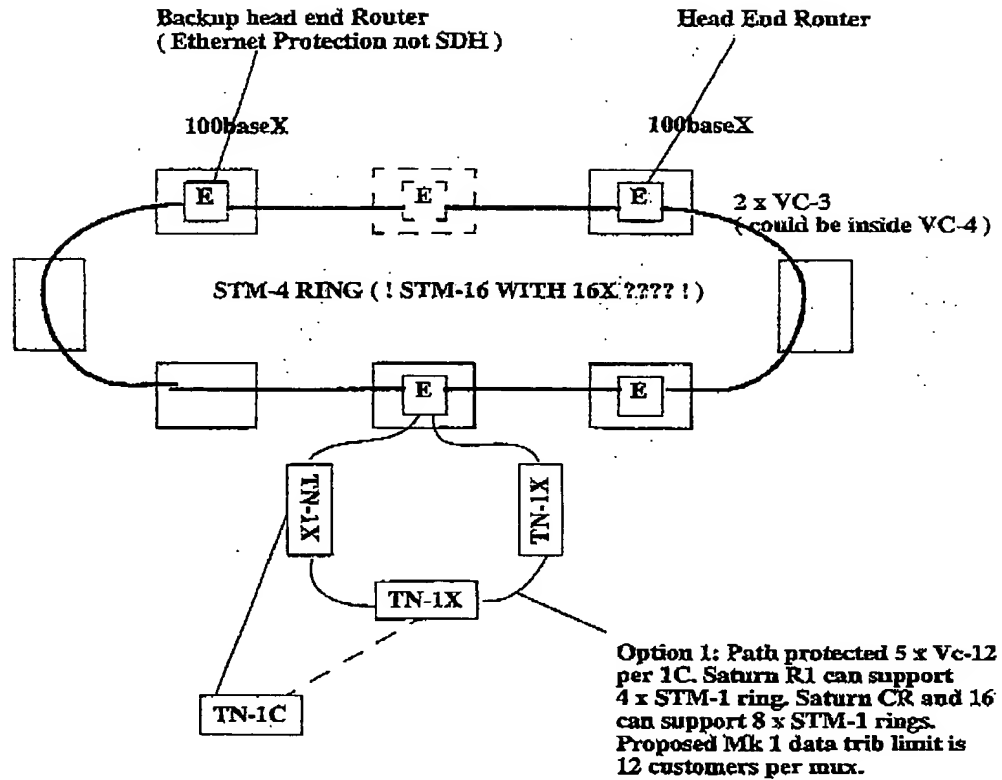
- RINGS ON CUSTOMER
PREM ARE A BAD IDEA.

- SINGLE CARD 1C IS
NOT REALLY A RING
PRODUCT.

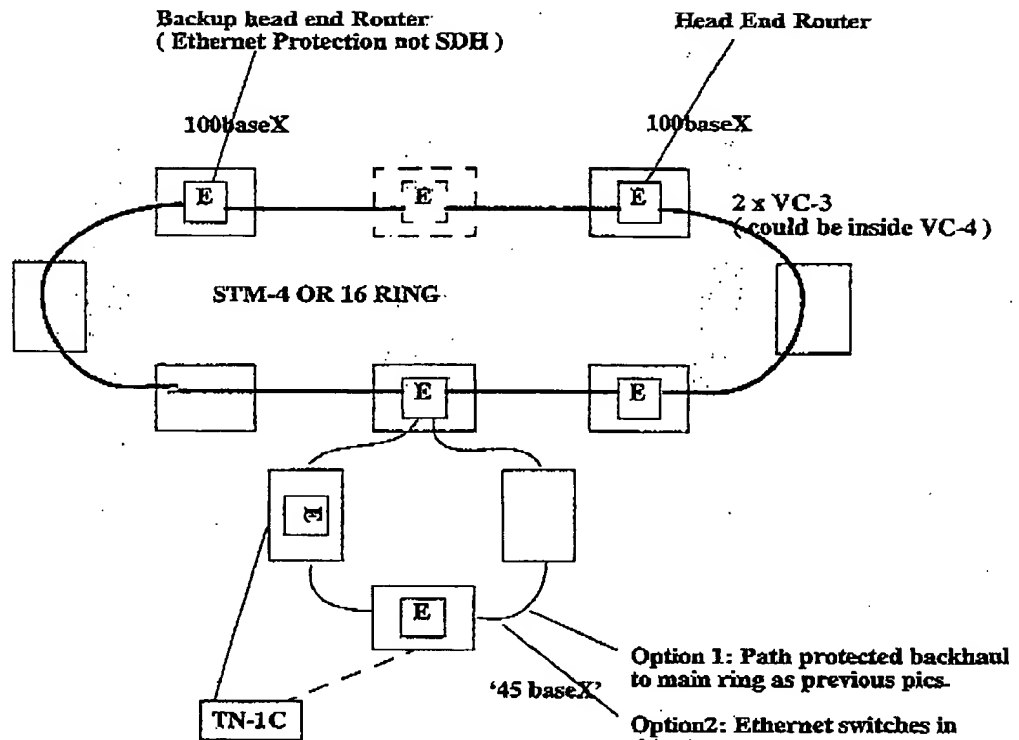
Option 1: Path protected 5 x Vc-12
per 1C. Saturn R1 can support
4 x STM-1 ring. Saturn CR and 16
can support 8 x STM-1 rings.
Proposed Mk 1 data trib limit is
12 customers per mux.
Option 2: 5 x Vc-12 per ring. 1C
does frame add/drop. Data trib
limit proposed : 6 rings.

Notes on Ethernet in Saturn

STM-1 RING ACCESS cont...



STM-4 RING ACCESS

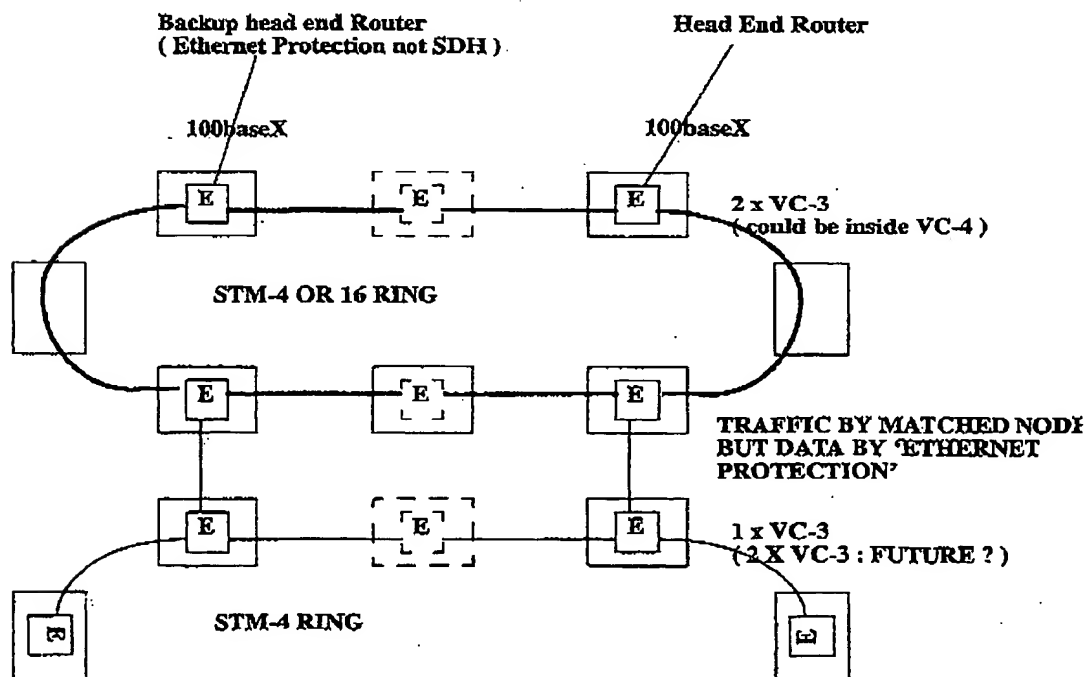
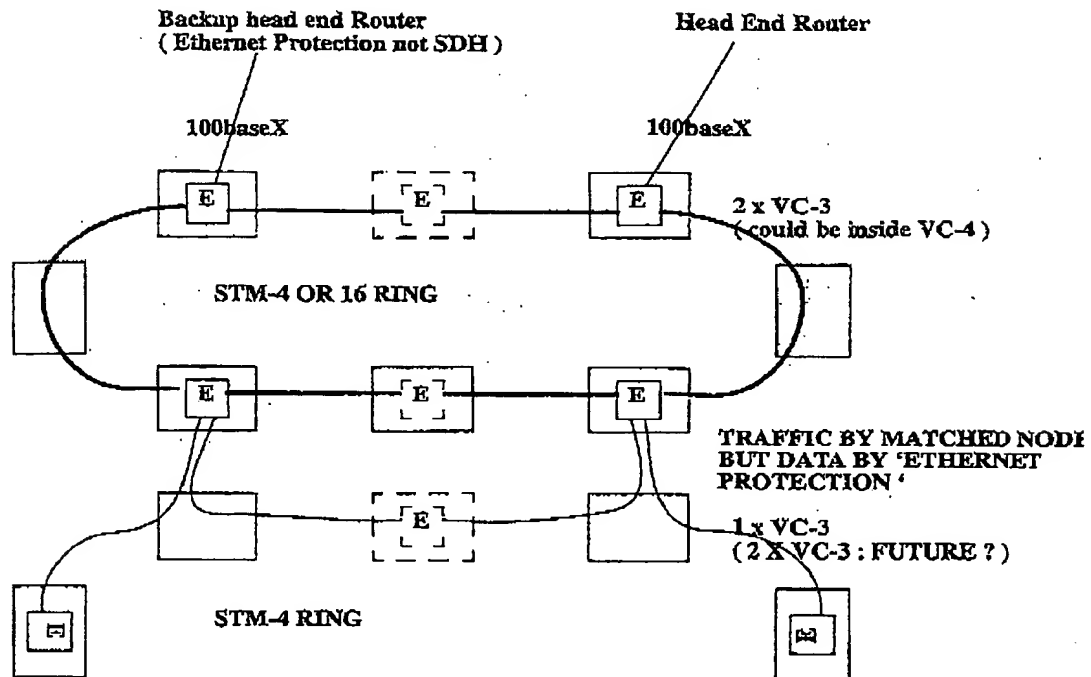


Option 1: Path protected backhaul to main ring as previous pics.

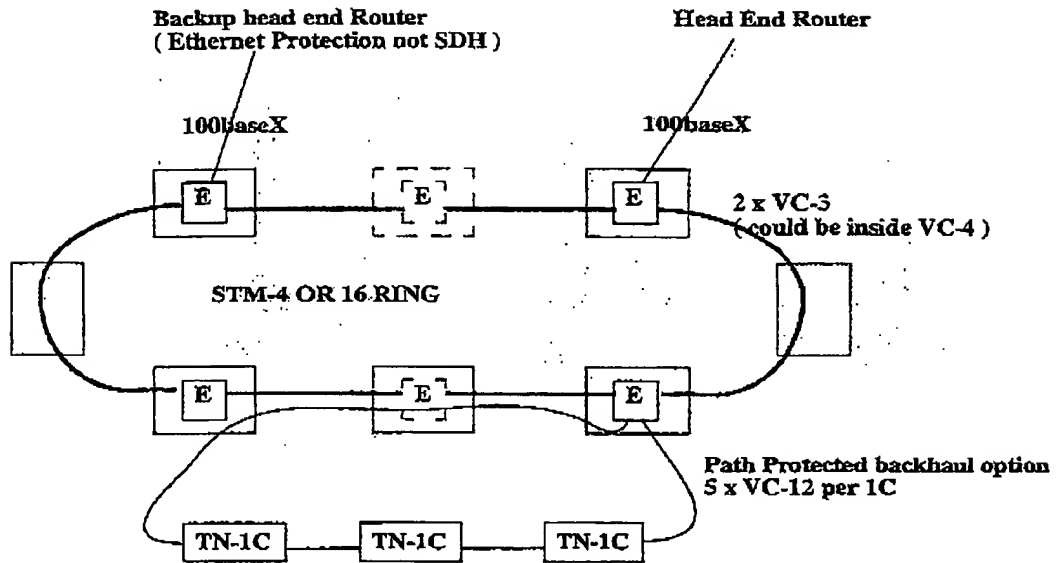
Option 2: Ethernet switches in this ring.
Assume Saturn CR can support 2 x STM-4 trib rings. Could allocate 1 x VC-3 per trib ring for Ethernet. Need 4 x VC-3 trib ports on Main ring datatrib. Saturn 16 may be able to support 4 x STM-4 trib ring. If so, datatrib would need 8 x VC-3 trib ports. This would leave up to 4 x VC-3 for main ring data bandwidth.

Choice of running subtending rings at 1 x VC-3 for data, seems to fit. But may need Mk2 data trib to fully exploit Saturn 16. Need 4 x VC-3 trib ports on Mk1 datatrib.

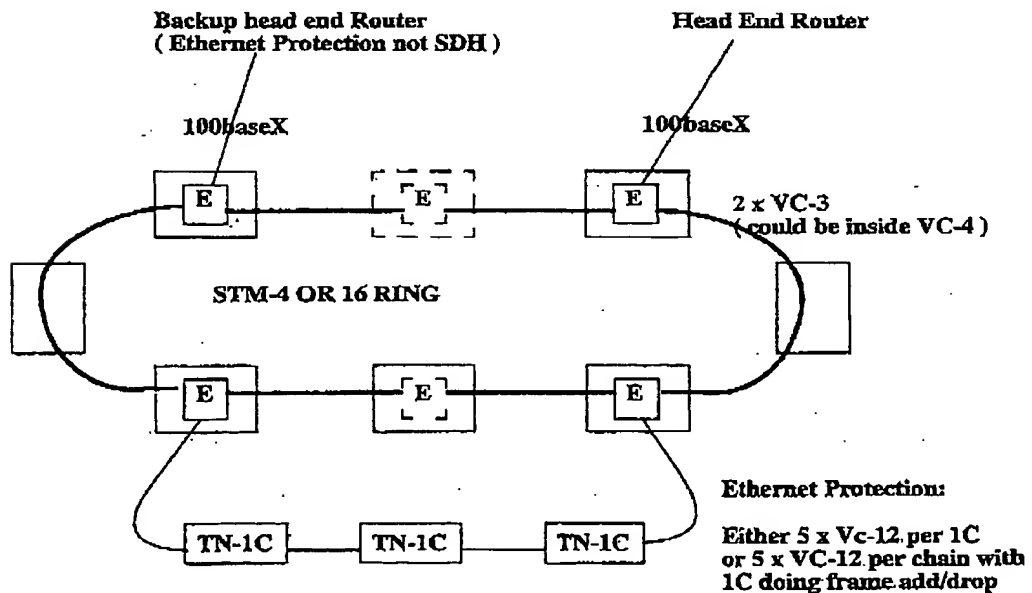
WHAT ARE WE GOING TO DO ABOUT MATCHED NODES ? (ACCESS SIDE)



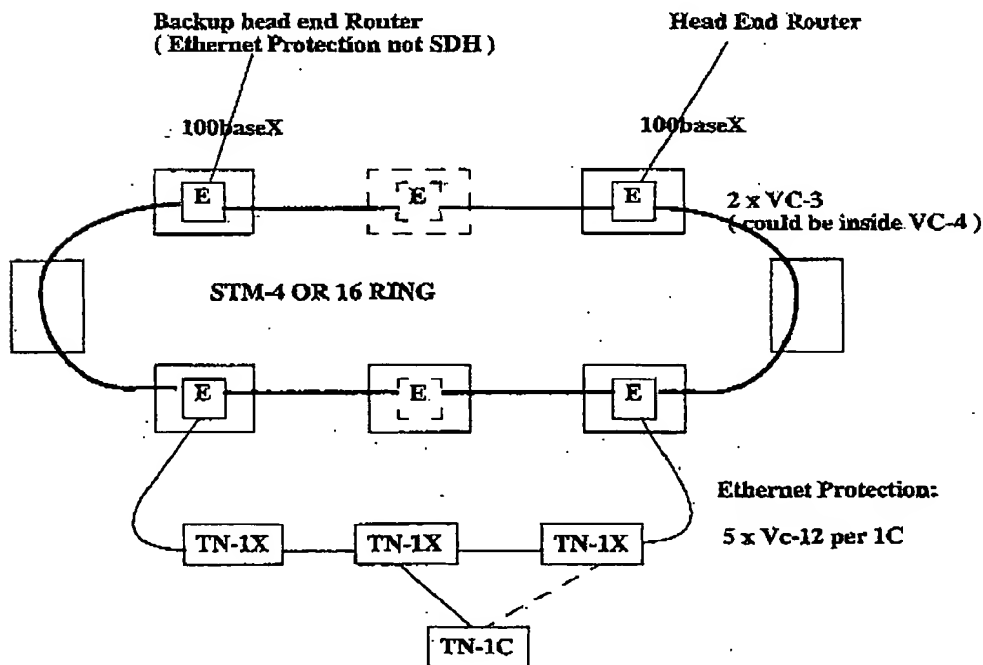
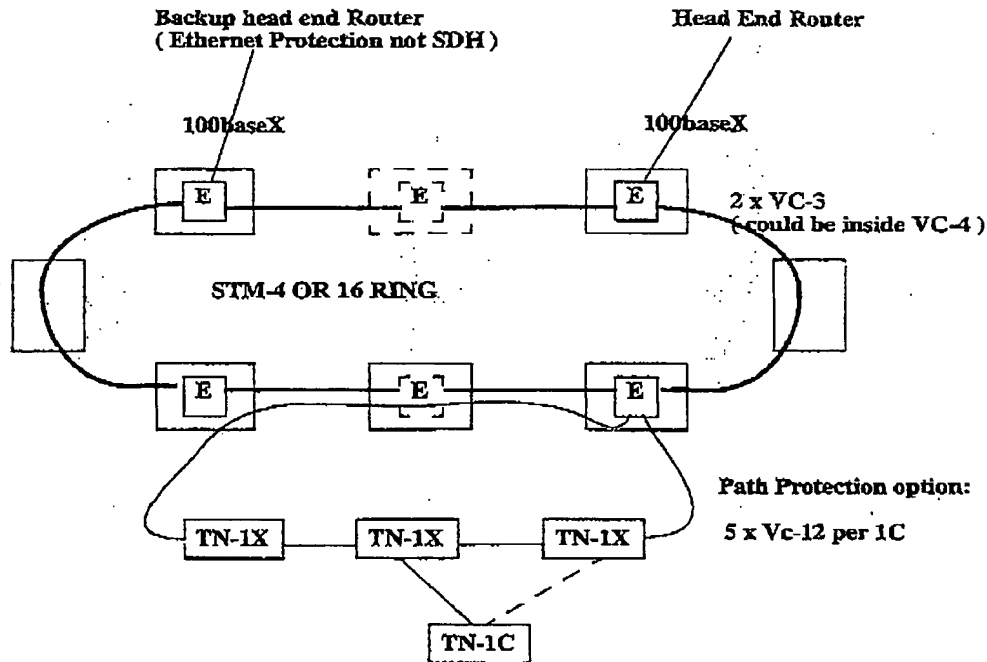
STM-1 ACCESS ARCS



BOTH PICS ASSUME 1C IS CUSTOMER PREM LOCATED. VIEW ON THIS EXPRESSED EARLIER.

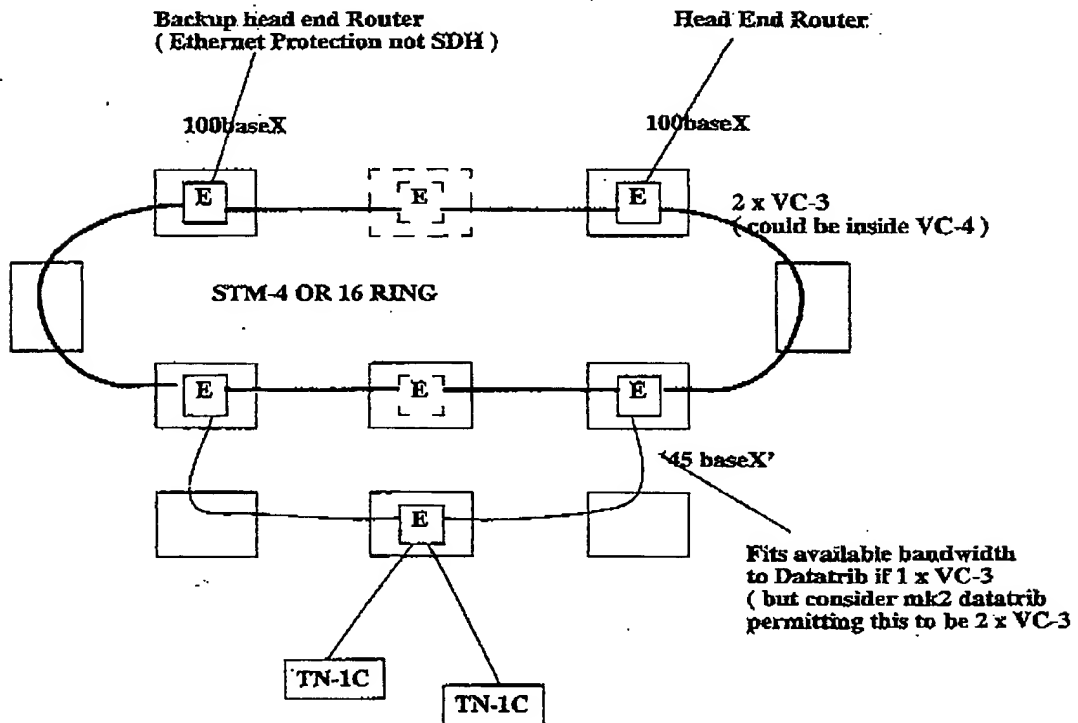


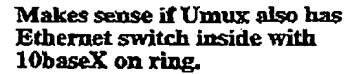
STM-1 ACCESS ARCS cont...



STM-4 ACCESS ARCS

FOR STM-4 MUXES WITHOUT DATATRIBS, JUST SUBSTITUTE FOR 1X IN PIC ON PREVIOUS PAGE.





THE STORY ISN'T SO GOOD FOR TWO CASES:

- CUSTOMERS LIKE CABLE COMPANIES, WHO HAVE ALSO DEPLOYED A LOT OF 1X, AND WHO WOULD LIKE TO OFFER ETHERNET TO SMALLER BUSINESSES.**

cont.. from previous pic.

SUGGESTION FOR 'COLT LIKE' CUSTOMERS

- IDEAL SOLUTION: NEW STM-1 MUX WITH ETHERNET SWITCH INSIDE
- ALTERNATIVE

ASSUMPTION, WE ONLY NEED A SOLUTION WHEN A LOT OF CUSTOMERS WANT ETHERNET.
THEREFORE ETHERNET MUST OCCUPY MINIMUM BANDWIDTH ON STM-1 RING.
THEREFORE 1C MUST HAVE ETHERNET INTO 1 X VC-12 OR INTO 2M OPTION.

IN A COLT TYPE SCENARIO, MOST ETHERNETS WILL BE INTRA-RING OR INTER RINGS OF THE SAME HEAD END.

SO SAY, 120 ETHERNETS OVER 4 STM-1 RINGS. ASSUME ABOUT 30 WILL NEED TO LEAVE THE HEAD END TOWARDS THE BACKBONE.

CONSIDER A HEAD END FLAVOUR OF SATURN DATA TRIB WITH A LOT OF ETHERNET PORTS ? (NEED ABOUT 30, EACH AT 1 X VC-12)

CONSIDER FITTING AN ETHERNET SWITCH INTO 1X, WITH MAYBE OPTIONS OF EITHER 5 X VC-12 OR VC-3 IN RING. ?

CONSIDER WHAT IS GOING TO REPLACE 1X ?

OF THE ABOVE, I THINK ETHERNET SWITCH INTO THE 1X IS PROBABLY HIGHLY DESIRABLE, BUT WOULD PROBABLY HAVE TO BE INTEGRATED ON THE PAYLOAD MANAGER. (PREFER TO SELL THEM A SATURN 4XE/S AS A REPLACEMENT)

SUGGESTION FOR 'CABLE TYPE' CUSTOMERS

- PROBABLY DO NOTHING - THEY WILL DEPLOY CABLE MODEMS
- OTHERWISE IDEAL SOLUTION IS THAT UMUX SITS IN RING AND HAS ETHERNET SWITCH
- ALTERNATIVE: UMUX CONSOLIDATES SEVERAL CUSTOMER DATA SERVICES INTO ETHERNET IN VC-12. BACKHAUL VC-12S, CONSIDER A HEAD END FLAVOUR OF SATURN DATA TRIB WITH A LOT OF ETHERNET PORTS ?

CONCLUSION ON THE ABOVE:

NO POINT DOING ETHERNET INTO SINGLE VC-12 OPTION ON MK1 SATURN DATATRIB, BECAUSE ITS ONLY NEEDED WHEN A LOT OF PORTS ARE ALSO NEEDED.

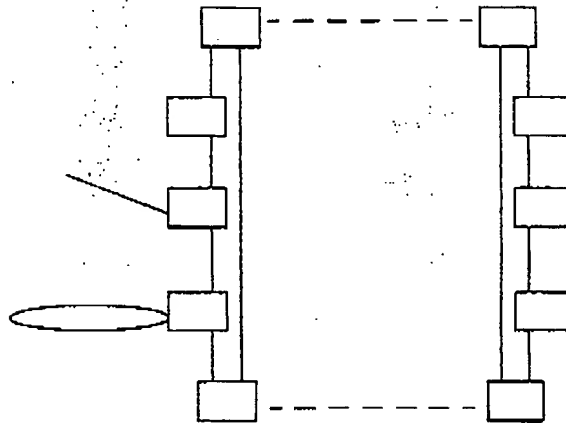
QUESTIONABLE IF ITS WORTH DOING ETHERNET INTO SINGLE VC-12 ON 1C

NEED TO WORK OUT 1X AND CABLE STREET CABINET STRATEGY BEFORE DOING ANYTHING ABOUT 'LOT OF PORTS' SATURN DATA TRIB.

Notes on Ethernet in Saturn

**FINALLY ON ACCESS (THERE IS A LOT MORE TO CONSIDER ON BACKHAUL
"UPWARDS" FROM THE SATURN RINGS.)**

ENERGIS-LIKE MESHES

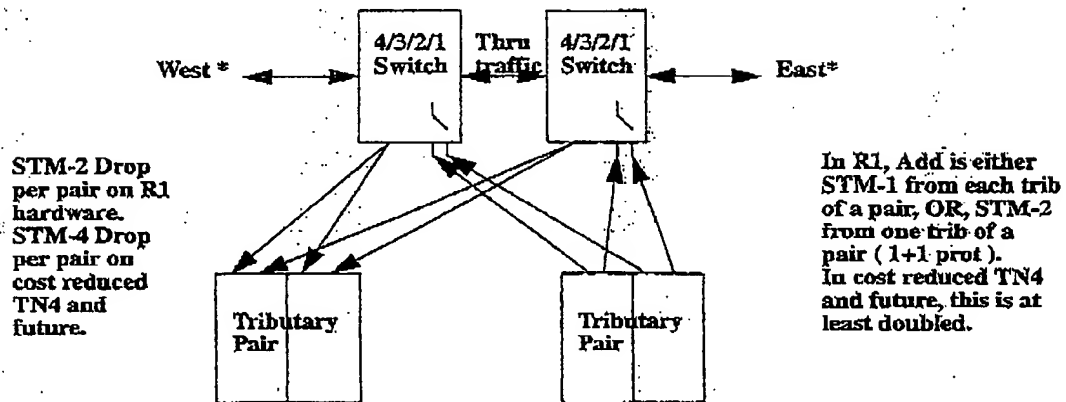


Energis actually deploy spurs or rings from their 'Snaps' to reach customers, so the Energis 'mesh' will be considered later as part of backhaul.

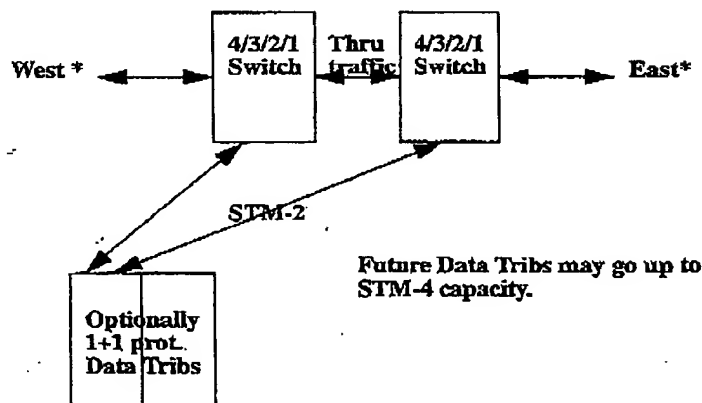
4.0 :IMPLEMENTATION IN SATURN

4.1: Saturn Routing Capability

* ADM versions

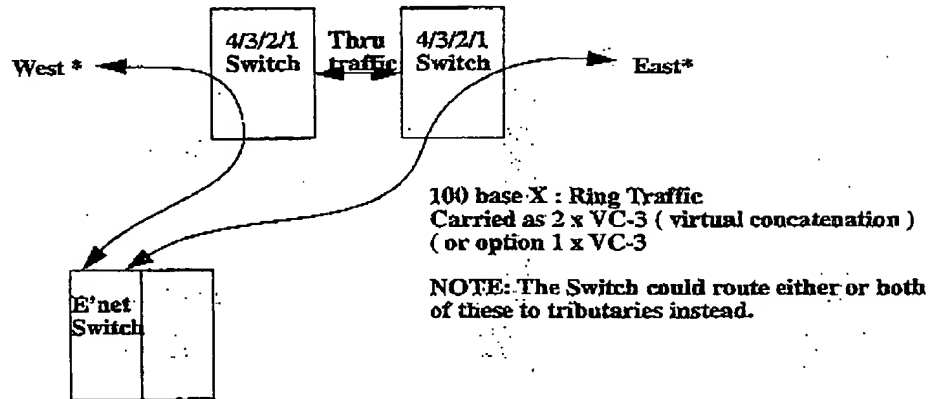


4.2: Data Option

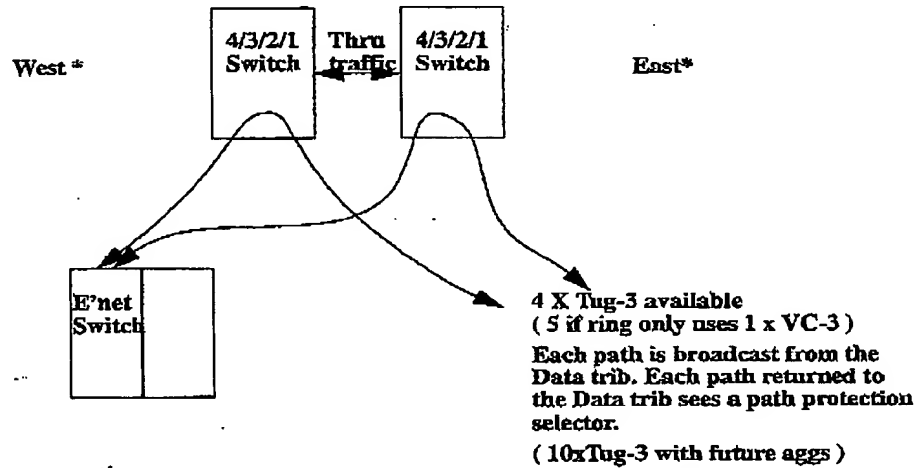


4.2.1: Use of the Data Trib

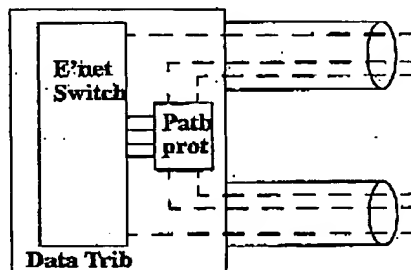
TRAFFIC WHICH IS NOT PATH PROTECTED



TRAFFIC WHICH CAN BE PATH PROTECTED



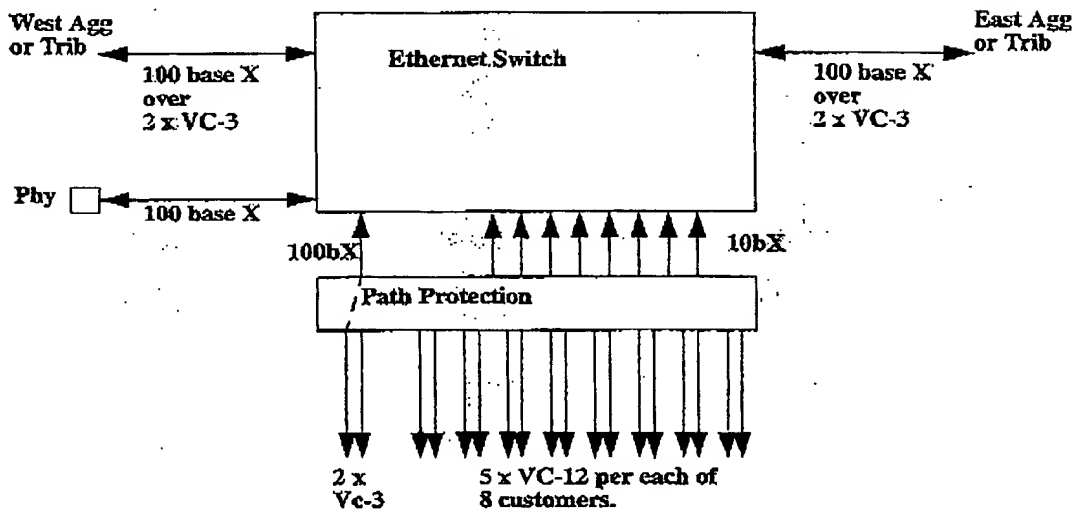
BACKPLANE UTILISATION



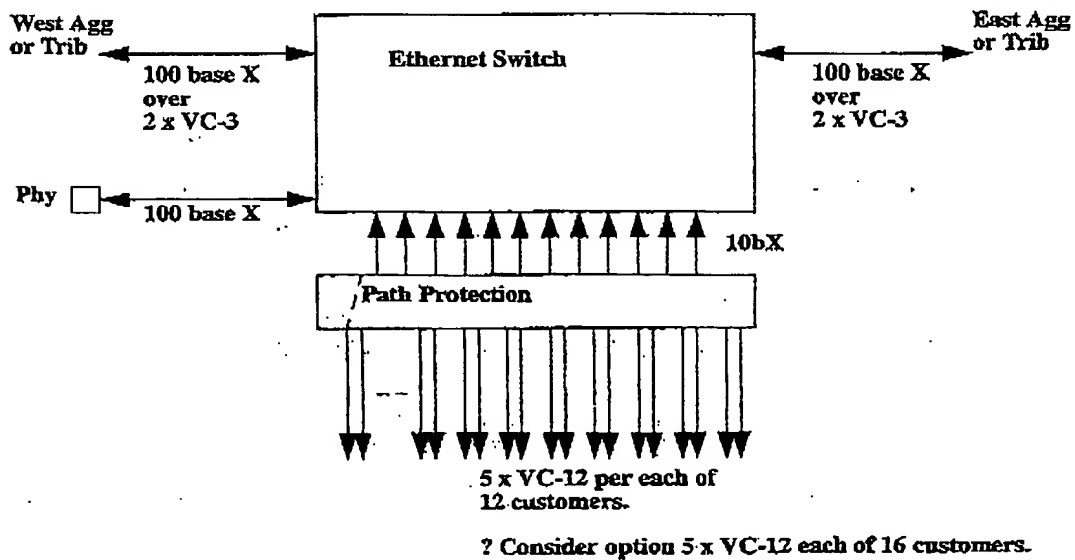
NEEDS FURTHER WORK

4.3: Modes of Operation

RING HEAD: Phy or Path Protected 2 x VC-3 access to Head End Router

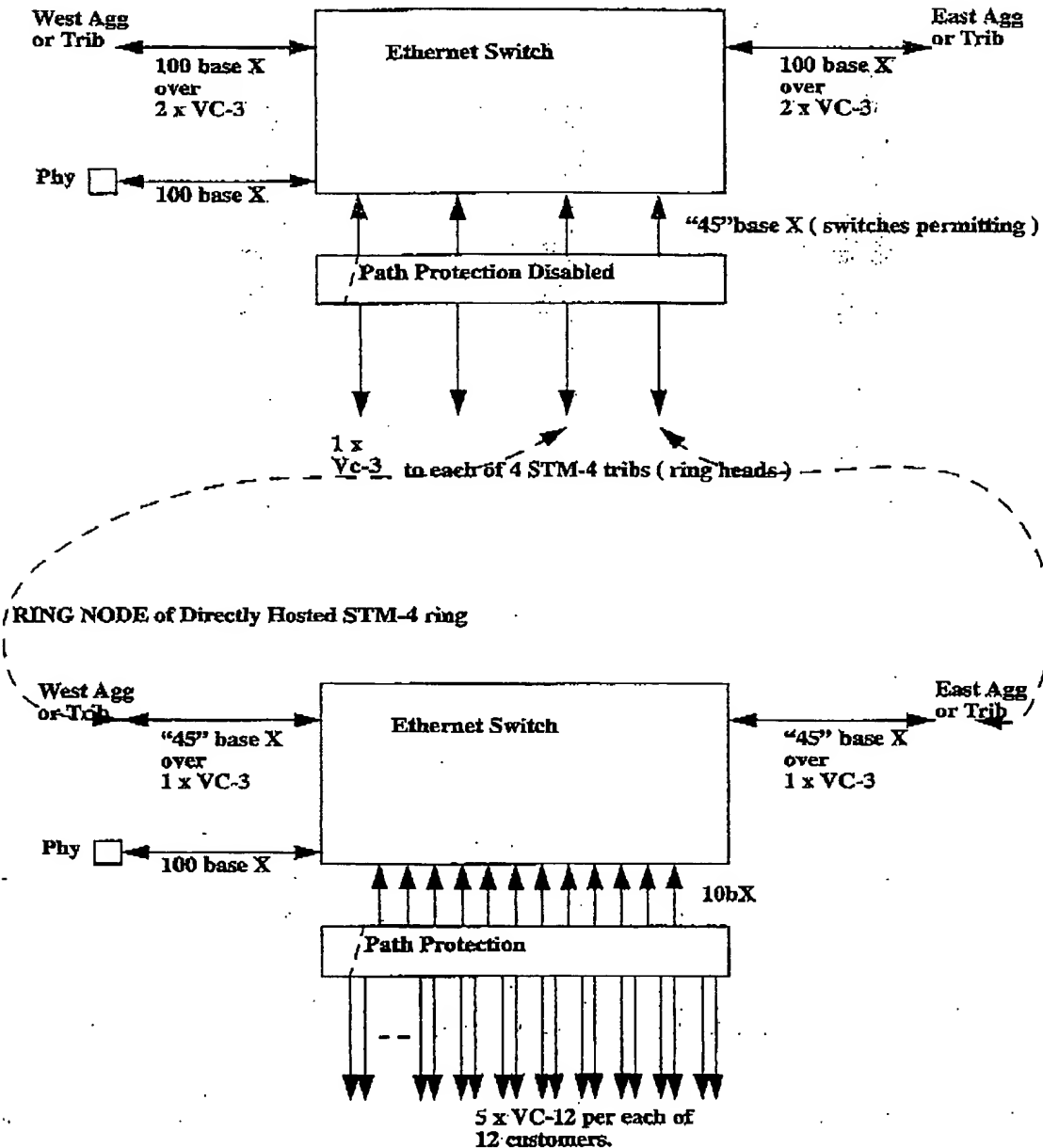


RING NODE with Fibre Spurs to CPE (or directly hosted STM-1 rings with 10bX in ring)



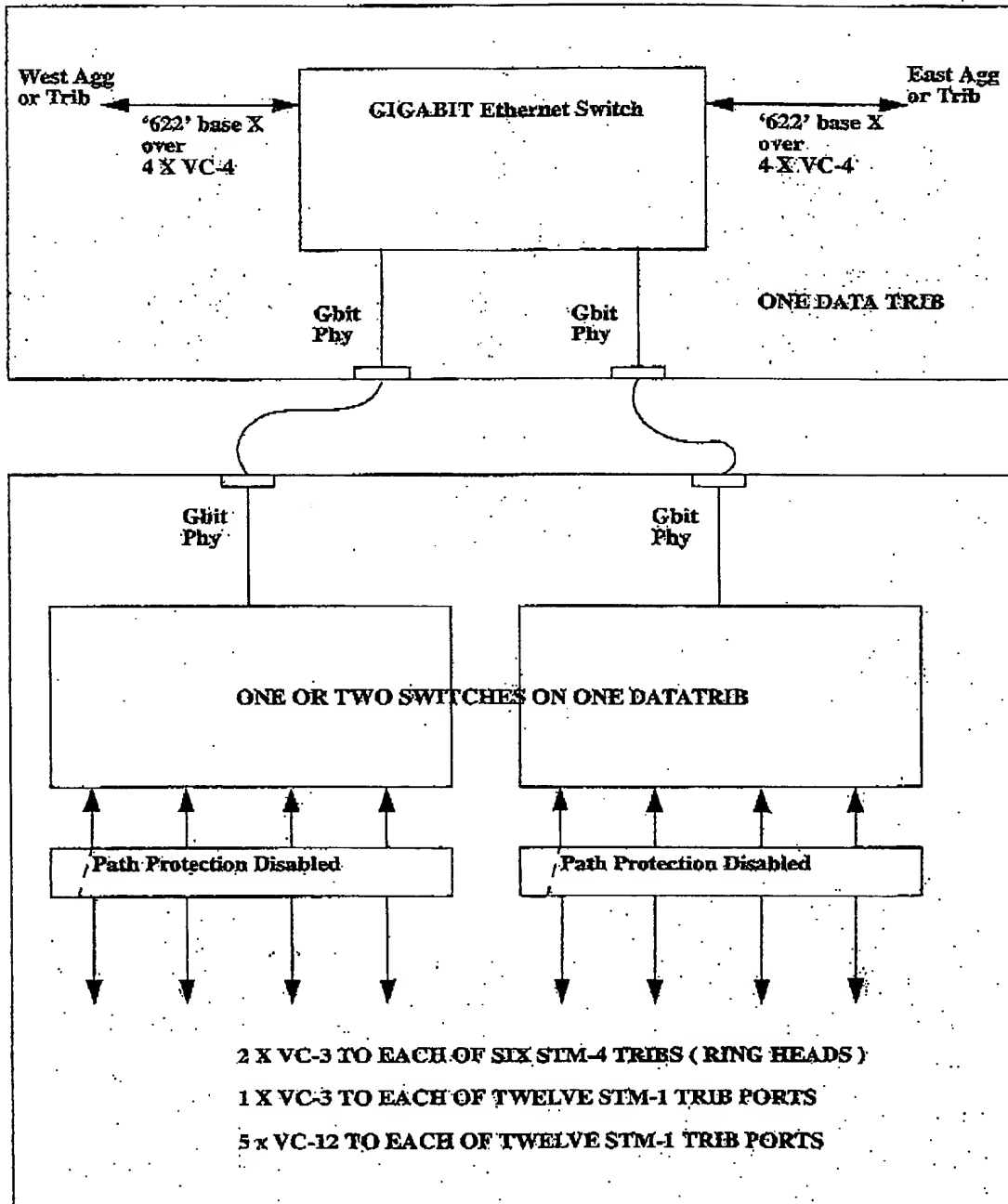
Notes on Ethernet in Saturn

RING NODE with Directly Hosted STM-4 rings. (max 2 rings)



THIS EXAMPLE APPLIES TO A SATURN 16 RING WITH SUBTENDING STM-4 RINGS
 THE INTENT IN THE LONGER TERM WOULD BE IN TWO STEPS. STEP 1: DATA TRIB
 WITH STM-4 CAPACITY AND THEN INTEGRATED DATA AGGREGATE. THE LATTER
 WOULD SUPPORT GIGABIT ETHERNET ON THE STM-16 RING

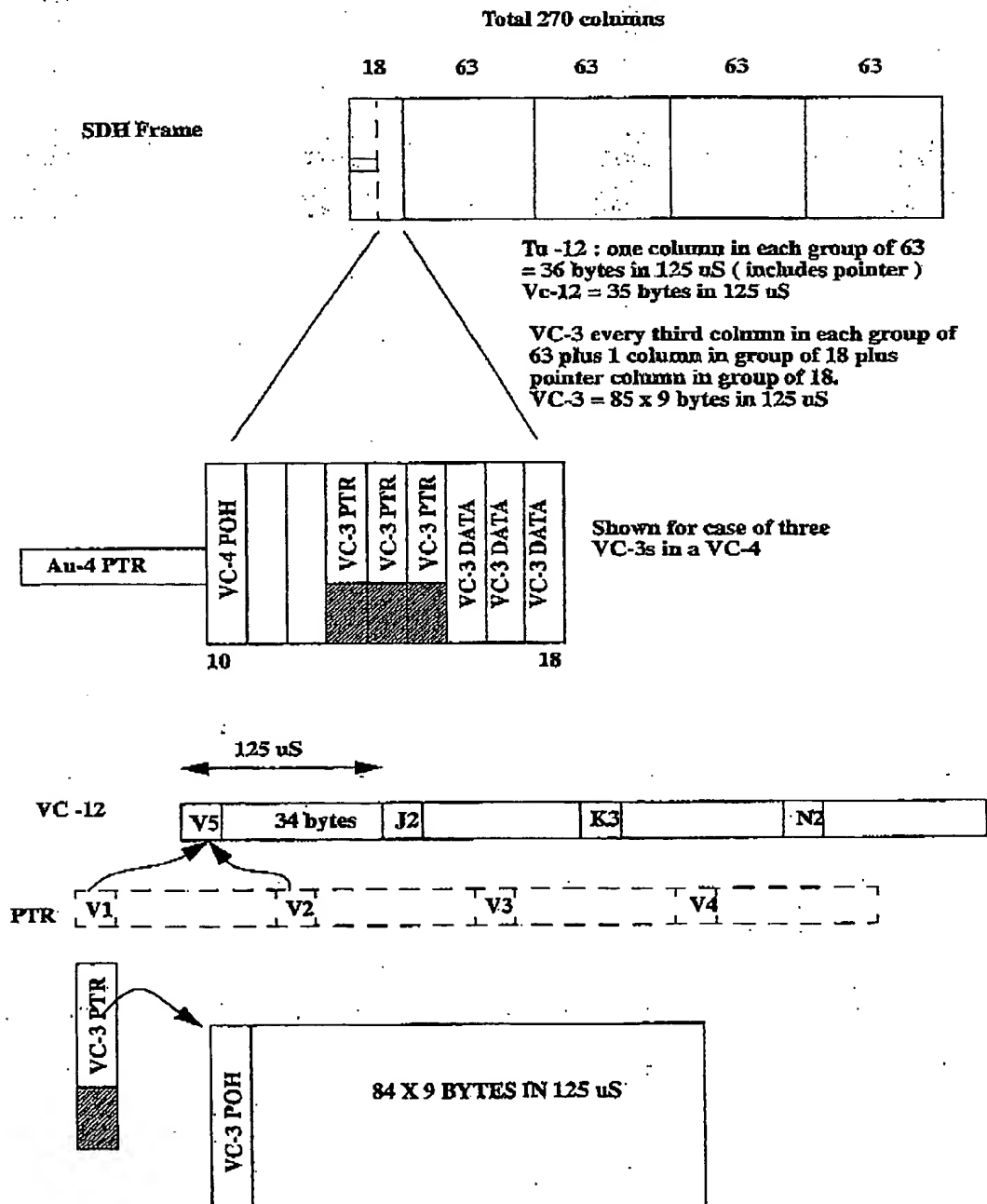
UPGRADE OPTION ON PREVIOUS PIC: (2 SLOTS UNPROTECTED)



5.0 :VIRTUAL CONCATENATION & FRAME PACKING

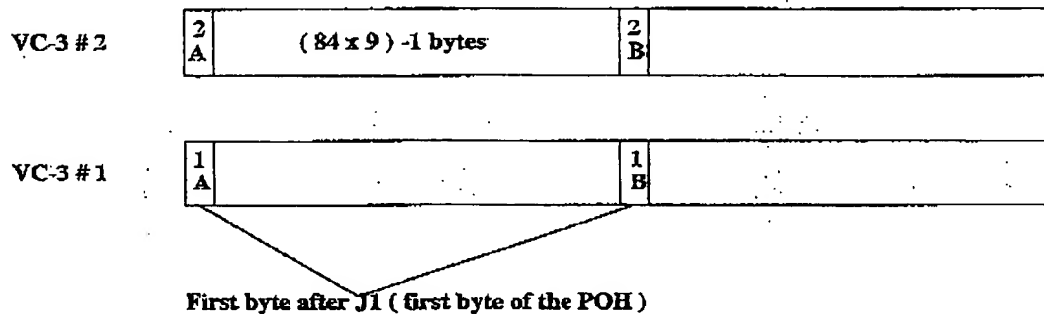
5.1: Virtual Concatenation

5.1.1: VC-12s and VC-3s



5.1.2: Virtual Concatenation: Originating End

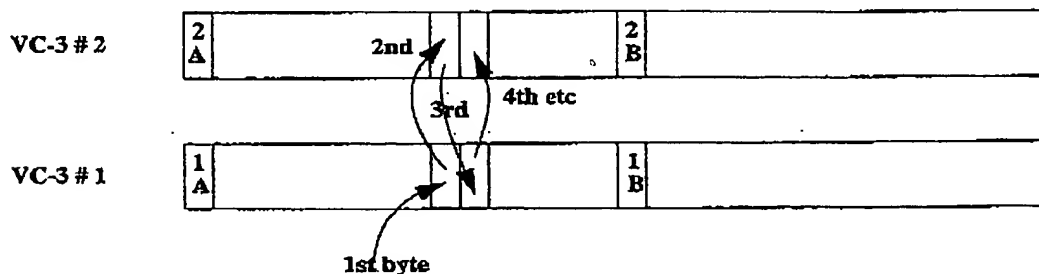
To describe the principle of virtual concatenation, the example of concatenating two VC-3s will be used. Firstly at the originating end, two VC-3s are created to carry the payload. They will be created with timing to suit the local equipment multi-frame sync and probably with a convenient pointer value. Both will be created 'simultaneously'. Only the payload region is illustrated.



One byte of the payload at a known location is used to append a frame number. The number of frames this number must increment over before repeating is determined by the differential delay the two VC-3s may incur crossing a network. Nominally if the differential delay is N frames, then the numbers must run over $2N$ frames before repeating. In practice some allowance must be made for the payload bytes not being uniformly spread over the 125 μ s interval and the number may have to run over one more frame than nominal.

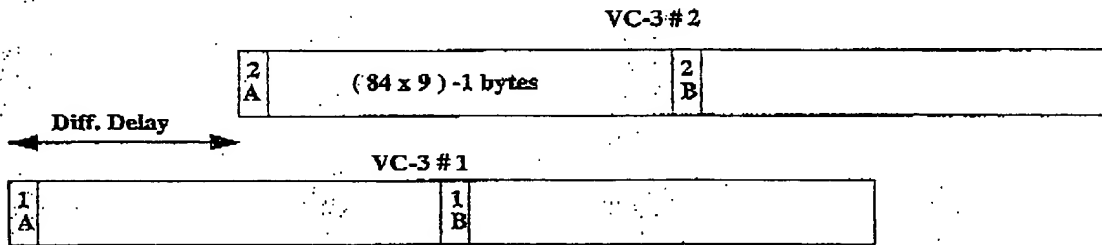
Differential delay can have two causes. For two VC-3s which run over the same physical route (fibre), the differential delay may be caused by intervening NEs. This will be null for NEs which manipulate only VC-4s (if the two VC-3s are in the same VC-4), otherwise it will be due to variations in fill of LO ptr processor buffers. (Any time switching should only add 1 byte delay - unless its a very badly designed time switch). For two VC-3s which run over different routes (which could happen if a path protection switch only occurs on one VC), then the differential delay is due to different physical distances (fibre delay) plus the absolute delays of intervening NEs.

A data frame to be carried in the concatenated payload can start at any time relative to the VC. It is not desirable to incur delays and storage in holding the data frame to the next VC start. It is inserted into the payload as shown (but note that encoding is covered later).

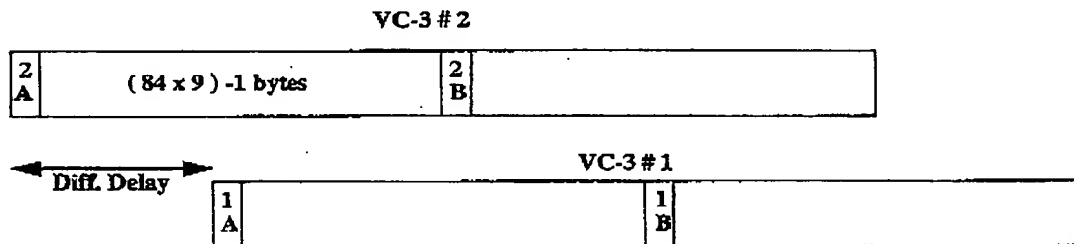


5.1.3: Virtual Concatenation : Destination End

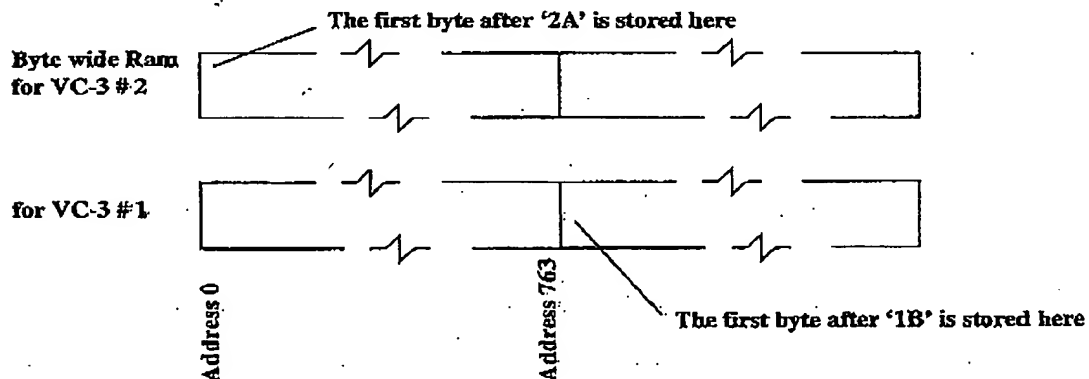
When the two VC-3s arrive at the destination equipment, they will have (in the case of Saturn) been pointer processed. Their pointers will therefore be at known locations, but the VCs will have suffered different delays. This is illustrated below, assuming that the delay is less than 125 μ s (for ease of illustration).



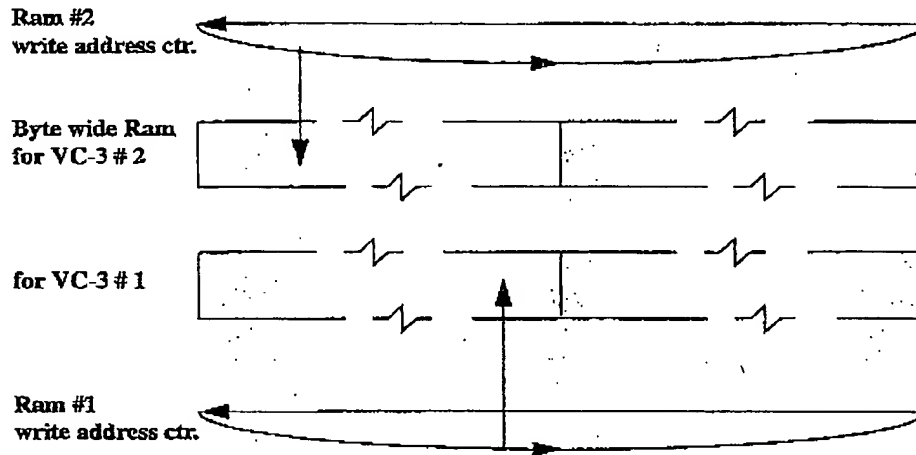
They could arrive as shown above,
or as shown below.



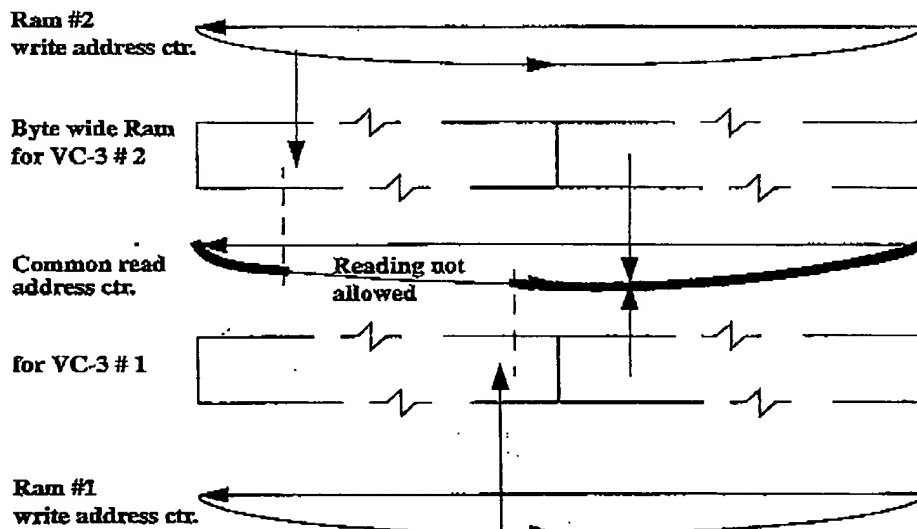
A pointer receiver is necessary to locate the payload, the payload is then identified by the following byte as 'A' or 'B' and it is written to a ram. The ram is organised as follows:



Assume VC-3 #2 arrives after VC-3 #1. At an instant in time the locations illustrated below will be being written.



The tricky bit is to control a read counter (this is rather like a pointer generator), so that simultaneous reading of both Rams occurs in a legal region illustrated below:



On the face of it, to control the read address counter, one waits for the last VC of the two to arrive (when it writes to address 0). The read address counter can then trickle along behind the write address of #2 above, so long as it doesn't fall so far behind that the write counter of #1 catches up with it. The problem is that if the system started up, just after #1 had written to address 0, it would think #2 was the first arriving frame and #1 arrived later. This is why the

frame number has to run for > twice the designed differential delay. If 'last' arrives more than the designed value after 'first', the read address counter needs resetting. Controlling the read address counter in normal operation should be hitless, so either a phase locked loop could be used or a clock gapping scheme. In either case a control circuit based on after 'last' and before 'first' is needed. The decision will be influenced by the nature of the encoding used to carry Ethernet data.

Note: it may not be necessary for the payload ram to be twice as large as the differential delay, but it's a lot easier to think about if it is.

Note: if the virtually concatenated VCs suffer a differential delay larger than the range of the frame numbering, this can only be detected by corruption of the payload data.

Note: One byte shown for frame numbering is illustrative. With one bit to indicate the VCno (not strictly necessary but a good check), this leaves 7 bits for frame sequence. This may or may not be enough.

Note: The choice of a frame number byte with its location determined by its place in the VC structure precludes the passage of concatenated payloads over PDH links.

5.2: Ethernet Encoding

5.2.1: The MII interface

(see IEEE 802.3 100baseT)

Assumption: We are only interested in connecting (via SDH) an Ethernet switch to an Ethernet switch, or an Ethernet switch to a Mac. We are not going to have an isolated Ethernet Phy (physical interface device) at the end of an SDH link.

Assumption: We will choose all Ethernet parts to support the MII standard. Older 10base X parts commonly used a 1 bit, 10M serial interface, but the MII seems to be becoming the norm. Its standard on 100base X and dual speed parts.

The MII is a 4 bit wide (nibble) interface clocked at 2.5 M for 10base and 25M for 100base. Depending on vendor chosen we may be able to use a gapped clock, but the maximum rate will probably be limited to 25M.

The Ethernet spec supports the following MII signals:

From a source:

(the transmit clock is input into the source)

- Tx_ER Transmit error (assumption : switches and macs do not source this signal ~ it can be used to cause corrupted symbols on the physical media)
- TxD<3:0> (assumption: switches and macs will not source corrupted frames, therefore sending a non-integer number of octets ~ odd number of nibbles ~ indicates a fault).
- Tx_EN Transitions high at the start of a frame (beginning at the start of the pre-amble) and remains high until the end of the frame.

To a destination:

(the receive clock is input into the destination)

- COL (Assumption: we are only operating in a full duplex environment, so collisions do not occur ~ any flow control is done by 802.2D Pause frame)
- CRS (Assumption: to be investigated if we need to locally spoof carrier sense to either macs or switches).

- Rx_D<3:0>
- Rx_DV Transitions high at the start of a frame (beginning at the start of the pre-amble) and remains high until the end of the frame. (Assumption: switches and macs rely on this signal to delineate the frame, they cannot recognise frame boundaries otherwise). Need to check that this is the same definition as Tx_EN, eg switches and macs do not put out an End of frame sequence which would be embraced by Tx_EN but not by Rx_DV.
- Rx_ER (Assumption: not used , its normally generated when a Phy detects a symbol error on the physical media , although we might consider using it if we lose a VC in the middle of a frame. -- Probably just rely on the switch detecting the bad CRC).

5.2.2: Encoding Constraints

There are two categories of choice to be made:

- The encoding of frame data and the delineation of frames.
- The clocking scheme to be employed.

Encoding Schemes:

Ethernet 'Phys' can detect start and end of frames by the presence of (usually) Manchester encoded data on the physical media. This is not possible for switches and macs, so when we transport an Ethernet frame over the SDH VCs, we need to somehow carry the information that this is a data frame as distinct from null or fill data. The Ethernet preamble cannot be used, because there is nothing to prevent user data from mimicking it. A number of schemes could be used:

- A simple segmentation and re-assembly scheme running over the payload of VC frames. The main objection to this, is that it is necessary to create VCs on a constant 125 uS or 500 uS clock. Therefore an Ethernet frame could be ready to transmit at any time during a VC frame. Simple segmentation schemes would require the storage of the Ethernet frame until the next VC start which could put up to 125 uS delay into the data transfer. With 100baseX, this would both delay and require storage of potentially several frames.
- A pointer based scheme. In such a scheme, a pointer in a known location of the VCs payload would point to the start of the Ethernet data frame. Another pointer (or a length field) would point to the end of the Ethernet data frame. Mechanisms would be provided to cope with frames which start and end inside one payload frame. This kind of mechanism is employed in ATM adaptation layers for AAL1 (but this assumes a regular structure - length ~ to the payload). There are two difficulties with this: Multiple Ethernet frames can fit inside a VC-3 payload (you could get a couple of minimum frames into a single VC-12 multiframe), so the number of pointers could be large and complex to use. You can't fill in the pointer value, until an Ethernet frame starts, so in order not to incur delay, the pointers have to go at the end of the VC frame and point backwards, this then incurs delay and storage at the destination while waiting for the pointer to arrive. (the storage might be incorporated in the storage required for 'de-concatenating', but the delay is probably unavoidable).
- A bit stuffing scheme. In this kind of scheme, the intent is to be able to recognise a string of ones or a string of zeroes as an interval between frames, by ensuring this doesn't occur in the data frame. Once it is known that no more than N (say) 1s will occur in a row in the data frame, strings of ones can be used to indicate an interval between frames, and strings of ones with a zero can be used to indicate start and end of frame. Such a scheme is used with Hdlc, in which any occurrence of a string of five 1s in the data causes the sending end to

insert a 0. Detection of five 1s at the receiving end causes removal of the following zero, or else detection of a special symbol like end of frame. This kind of scheme appears less efficient in use of bandwidth than pointer or segmentation type schemes, but this might be illusory. Pointers and segmentation carry a penalty in bytes used, also efficiency is about throughput and delay. Since with bit stuffing type schemes, both transmission and reception can begin as soon as the first data byte arrives, they might well minimise total delay. A simplifying variant on bit stuffing could be considered. Conceptually the data stream running in the payload of the VCs could be considered as 9 bit bytes starting after the first byte of path overhead. The 9th bit could then be used to indicate the presence of a frame. The inefficiency of an Hdlc like bit stuffing mechanism depends on the nature of the data in a data frame, the 9th bit scheme would be worse than best case Hdlc and better than worst case Hdlc. It might be a lot simpler to implement because the data rate over the VC payload is calculable.

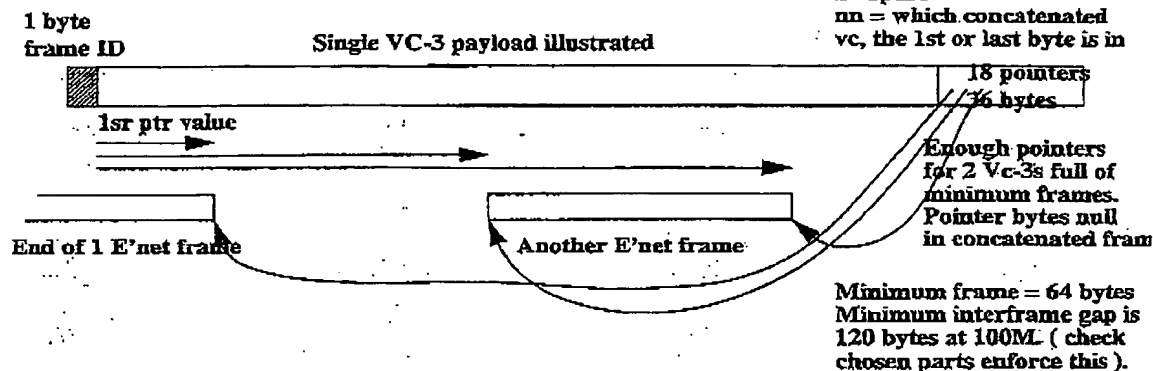
- A byte stuffing scheme as per PPP over Sonet (RFC1662). This is Hdlc but with the flag character (7E) and the abort character (7D) being replaced by 7D,5E and 7D being replaced by 7D,5D. The use of abort would permit 'forwarding' the 'error' signal equivalent of the MII. This scheme is attractive in its simplicity.

Clocking Schemes

Coming from a transmission background, it is natural to assume that data transmission from one Ethernet part to another over SDH will be continuous, with recovered clocks used at every stage of the process. This need not be the case. Since data frames are discrete, and all clocks are very accurate, it may be possible to use fixed clocks at different stages and small fifos to cater for differences. For instance, a gapped clock from SDH transport might be used to write concatenated VC-s into a Ram at the destination end. A fixed local clock might be used to drive the read address counter, with the read address being reset in the gaps between data frames. In either case, care will be needed to ensure that Ethernet inter-frame gap times (? 9.6 uS) are not violated (although a switch or mac might not care ~ which would make our life easier).

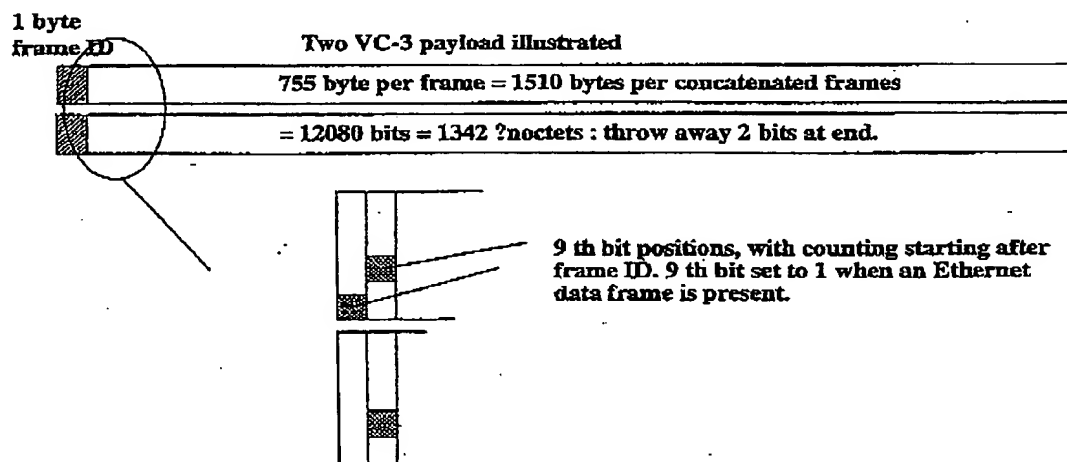
Some examples of the above schemes are illustrated below:

Pointer Scheme in concatenated VC-3 Frames.
 Payload = $(84 \times 9) - 1 \text{ byte} = 755 \text{ byte per frame}$.
 Need 10 bit pointer



NOTE: !! NEED TO CHECK IF INTERFRAME GAP IS NEEDED IN FULL DUPLEX SWITCHED SYSTEM, ALSO IF SWITCHES / MACS HAVE ANY CONTROL OF IT !!

9 Bit Byte scheme in a VC-3 payload



Final notes on this section:

The susceptibility of any scheme to bit errors needs to be considered. Since an Ethernet data

frame will be corrupted by bit errors anyway, the only issue is that the scheme doesn't lose sync and result in corruption of the next frame.

The choice of pointer or some form of bit stuffing scheme will probably be dictated by ease of hardware implementation. For efficiency, the pointer scheme might require the number of pointers used to be variable. eg a lot of short frames in a high rate payload would need a lot of pointers. Rather than reserving space permanently, it could be more efficient to run pointers from the back of the SDH frame forward until an 'end of pointers' marker.

5.2.3: Resultant Data Rates:

1 x VC-3 with pointer scheme: 84 columns - (1 byte frame Id for concatenation + 36 bytes of pointers) = 719 bytes in 125 uS = 46.016 Mbit/s. (so 2 x VC-3 = 92.32 Mbit/s). This might reduce slightly if more than 1 byte is needed for frame Id, and might be improved if the pointers were located across both concatenated frames.

1 x VC-3 with 9th bit scheme: 84 columns - 1 byte frame Id = 755 bytes in 125 uS. = 6040 bits in 125 uS = 6040/9 9bit bytes = 671 '9s' in 125 uS. This gives an Ethernet frame data rate of 671 bytes in 125 uS = 42.951 Mbit/s (so 2 x VC-3 = 85.9 Mbit/s).

An HdLc like scheme in which the end of frame flag is ignored , the preamble is used for start of frame, and no bit stuffing occurs, would result in 755 bytes in 125 uS, which is 48.32 Mbit/s (or 2 x Vc-3 gives 96.64 Mbit/s). This would go down to 80 Mbit/s with worst case bit stuffing.

1 x VC-12 with pointer scheme: 136 bytes - (1 byte multiframe Id for concatenation + 6 bytes pointers ~ 3 ptrs) = 129 bytes in 500 uS = 258 kbytes/s = 2.064 Mbit/s .
So 5 x VC-12 gives 10.32 Mbit/s.

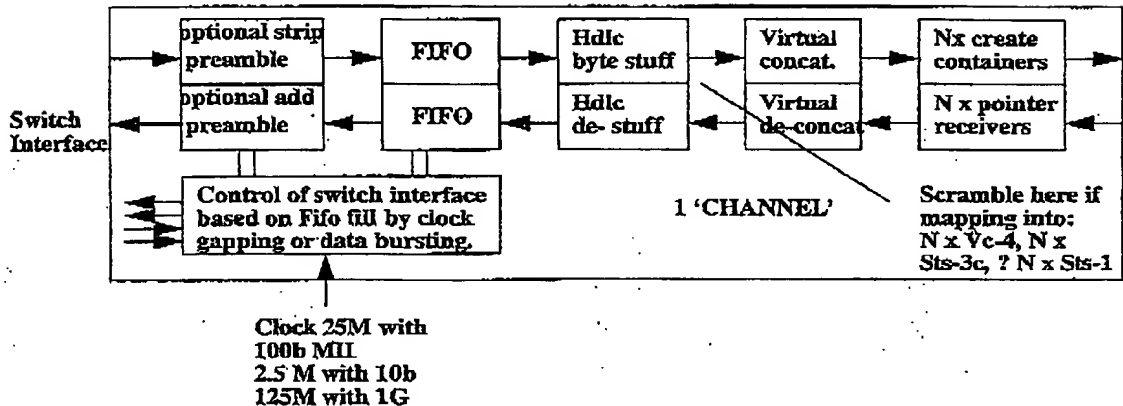
1 x VC-12 with 9th bit scheme: 136 bytes - 1 byte multiframe Id for concatenation = 135 bytes in 500 uS. = 1080 bits = 120 9bit bytes in 500 uS. This gives an Ethernet frame data rate of 120 bytes in 500 uS = 1.92 Mbit/s. So 5 x VC-12 gives 9.6 Mbit/s.

TO BE CALCULATED FOR HDLC BYTE STUFFING

5.2.4: Scrambling

If the user data frames can fill an SDH section (eg: data mapped directly into a VC-4 sent over an STM-1 link), it is possible for the user to maliciously mimic the SDH scrambler sequence. This could result in all 0s or all 1s data across the link which would cause the section clock recovery PLL to fail. Therefore scrambling using a self synchronous scrambler has been proposed to prevent this.

6.0 :BUILDING BLOCKS



THE ABOVE SEEMS TO BE A COMMON BUILDING BLOCK FOR MOST APPLICATIONS, BUT IT MIGHT BE IMPLEMENTED IN VARIOUS VERSIONS FOR DIFFERENT RATES, WITH OR WITHOUT VIRTUAL CONCATENATION AND WITH OR WITHOUT SCRAMBLER, POSSIBLY LO OR HO PTR RECEIVERS.

NOTES: THE CLOCK ON ETHERNET SWITCHES CAN BE GAPPED, BUT CAN'T RUN FASTER THAN NOMINAL. THEREFORE THE CHANNEL DATA RATE ACTUALLY SENT OVER THE FIBRE MUST BE LESS THAN 1GBIT (100 MBIT) (10 MBIT). (NOTE : CHECK MPC860 USED ON ACCESS DEVICE)

OPTIONALLY STRIPPING PREAMBLES (ETHERNET), IS TO MAKE THE FRAMES ON THE FIBRE LOOK STANDARDS COMPLIANT (IF NECESSARY), NOT TO IMPROVE DATA RATES. INTO AN ETHERNET SWITCH, PREAMBLE AND INTER-FRAME GAP MUST BE PRESENT.

ITS ASSUMED THAT FRAMES AT THE SWITCH INTERFACE ARE COMPLETE WITH CRC-32.

HDLC STUFFING SENDS FLAGS BETWEEN FRAMES.

VIRTUAL CONCATENATION WILL BE CHALLENGING AT HIGH RATES. AT STM-16 / STS-48, A RAM FOR DE-CONCATENATION WOULD NEED A 600 MBYTE/S ACCESS AND 37.5 KBYTES PER 125 μ S DIFFERENTIAL DELAY. I THINK THIS MEANS A RAMBUS DRAM.

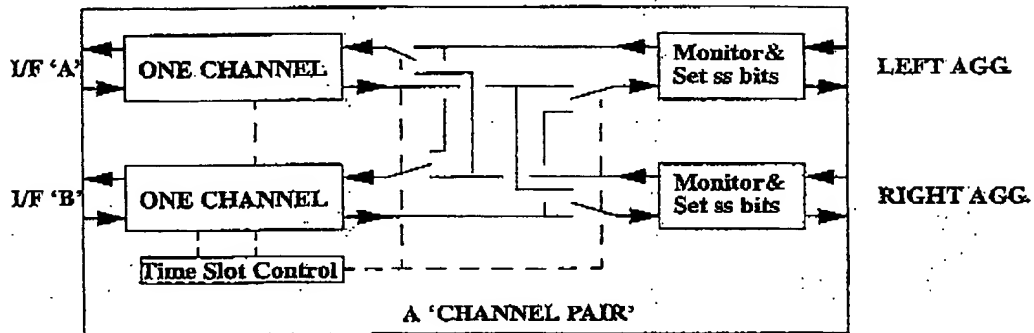
(NEED TO CHECK : IS POH POSITION AVAILABLE FOR REAL CONCATENATED SONET FRAMES. EG: VC-4 HOLDS 260 X 9 BYTES, DOES AN STS-3C LOSE 2 COLUMNS TO STUFF IN THE FORMER POH POSITIONS ?. DITTO N X VC-4)

FOR SATURN WANT LO PTR RECEIVERS CONFIGURABLE AS:
1-5 X VC-12 (? VT-1.5), OR 1-2 X VC-3 (1-3 X STS-1 HO IN SAME PART ?).

NEED TO CONFIGURE BY TIME SLOT, MODE AND SIZE

FOR LO PTRS, IT MIGHT BE POSSIBLE TO HANDLE VIRTUAL CONCATENATION AND POINTER RECEPTION WITH A PROCESSOR AND RAM FOR SEVERAL CHANNELS.

BUILDING BLOCK IN SATURN



SELECTOR POSITIONS ARE APPLIED FOR THE DURATION ONLY OF THE TIME SLOTS TO WHICH THE CHANNELS ARE ASSIGNED. THEY ARE SET BY CARD CONTROLLER SETTING REGISTERS.

THIS MEANS THAT:

EACH CHANNEL CAN BE ASSIGNED TO EITHER THE SAME AGGREGATE, BUT DIFFERENT TIME SLOTS, OR DIFFERENT AGGREGATES WITH THE SAME OR DIFFERENT TIME SLOTS, OR EACH CHANNEL CAN BE USED PATH PROTECTING ON BOTH AGGREGATES.

GENERAL IDEA IS THAT BACKPLANE CLOCK AND SUITABLY DELAYED BACKPLANE MULTI-FRAME SYNC IS APPLIED TO THE TIME SLOT CONTROL BLOCK. A PROGRAMMABLE OFFSET IS AVAILABLE FOR THE 'TO BACKPLANE' DIRECTION.

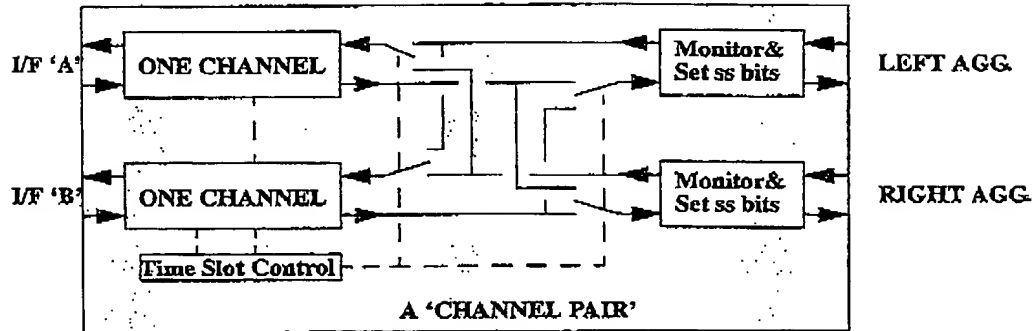
TIME SLOT CONTROL GENERATES STROBES ONCE EACH CHANNEL HAS BEEN PROGRAMMED WITH TIME SLOT, MAPPING AND AGGREGATE USAGE. THE STROBES RUN TOWARDS THE BACKPLANE TO CONTROL THE SELECTORS AND THE SS BIT MONITORING / INSERTION. (SATURN NEEDS BOTH, ? EXPRESS ONLY NEEDS INSERTION ?)

SS BIT INSERTION / MONITORING MIGHT BE BETTER DONE COMMONLY FOR SEVERAL CHANNELS.

FLEXIBILITY IS BEST IF THE ABOVE 'CHANNEL PAIRS' MUX TOGETHER INTO A STM-4 / STS-12 RATE BYTE STREAM, BUT IT MIGHT BE EASIER FOR IMPLEMENTATION IF MUXING IS BY TRI-STATING INTO A STM-1 / STS-3C BYTE STREAM FOLLOWED BY A 'ROUND ROBIN MUX' TO 78M.

BEFORE REACHING THE BACKPLANE, SYSTEM SPECIFIC SECTION AND LINE OVERHEAD (BYTE STEALING ETC) HAS TO BE CATERED FOR.

COMMON 'CHANNEL PAIR' FOR 1X, 1C, SATURN, EXPRESS



EACH 'AGG' PORT RUNS AT $78 \text{ M} \times (8 + \text{parity}) = \text{STS-12} / \text{STM-4}$

EACH SWITCH I/F RUNS AT : (by config register)

- MII 4 bits 25 M or 2.5 M clock

or

- GMII 8 bits clock at either 125 M if technology permits or (probably fpga) rate to suit STS-12 / VC-4 ~ 74.9 M (but would need to run a bit faster to permit the odd clock gap to keep the fifos under control).

MAPPINGS:

Either the 'Agg' interface has live STS / VC-4 traffic or else the HO containers have been synchronised elsewhere.

So either HO pointers have to be processed to extract data packets from $n \times \text{STS-1}$ or $n \times \text{VC-4}$ or else LO pointers have to be processed to extract data packets from $n \times \text{VT-1.5}$ or $n \times \text{VC-12}$ or $n \times \text{VC-3}$.

Higher and lower order pointers do not have to be processed simultaneously on the same channel.

Mappings supported (each can be real or virtual concatenation):

1~ 77 x VT-1.5 with 4 x 2.5 M MII

1~ 3 x STS-1 with either GMII or 4 x 25M MII up to 100 Mbit/s

1, 2 or 4 x STS-3c with GMII

1~5 x VC-12 with 4 x 2.5M MII (NOTE: Assume virtual concatenation within a Tug-3 ?)

1~3 x VC-3 with either GMII or 4 x 25M MII up to 100 Mbit/s

1, 2 or 4 x VC-4 with GMII

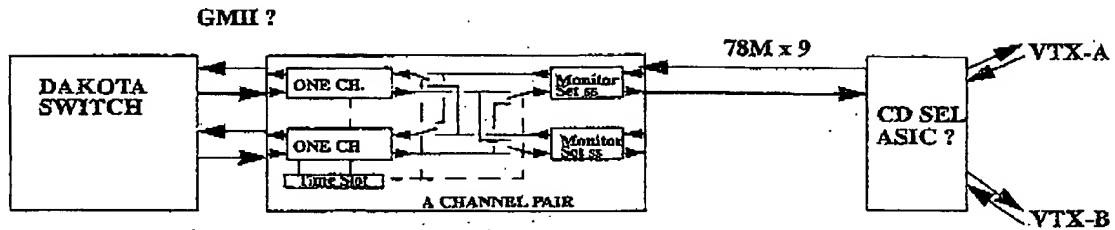
NOTE: Option with $n \times \text{STS-1}$ mappings to add fixed stuff to STS-1 to reduce payload to the same size as a VC-3.

NOTE: For 1X / 1C only $N \times \text{VC-12}$ is needed, but may need to program each VC to separate time slots. Or group of N within one Tug-3 ?

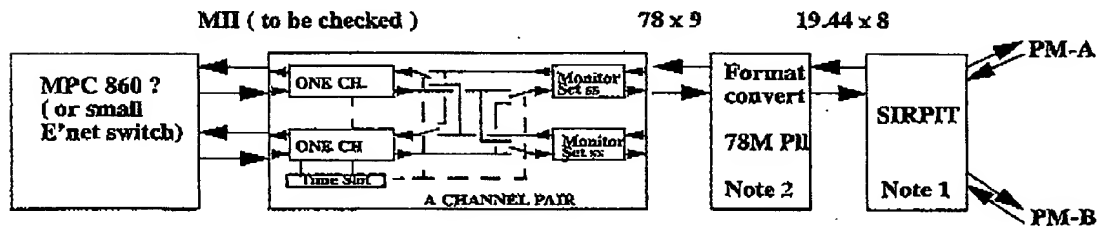
Notes on Ethernet in Saturn

APPLICATION WITH VARIOUS SYSTEMS

EXPRESS



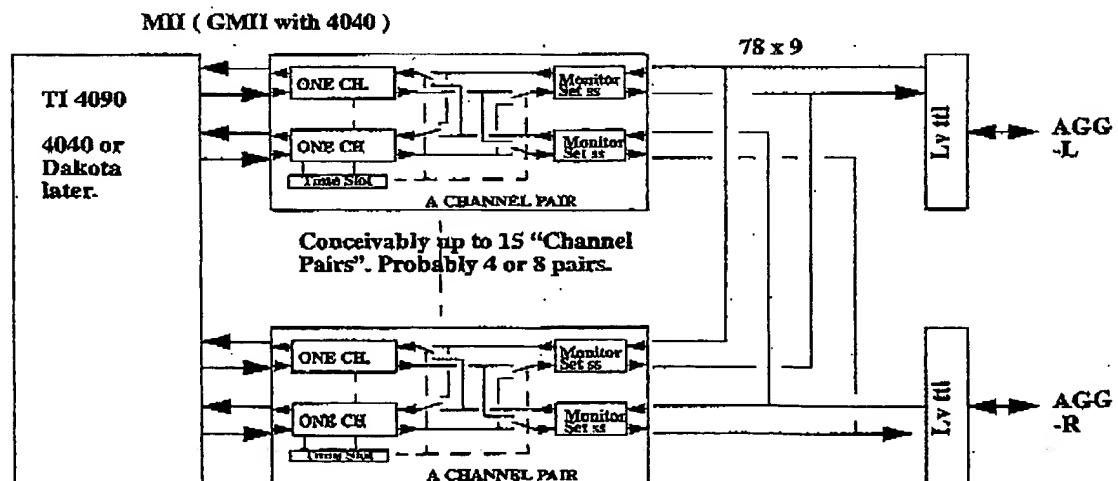
TN-1X / 1C



Note 2: bpa > "mapper"
just re-clock 19M data
to 78M.
"mapper" > bpa just
clock out STM-1#1 at
19M

Note 1: no Sirpit on
1C, byte parallel
back to only one "pm"

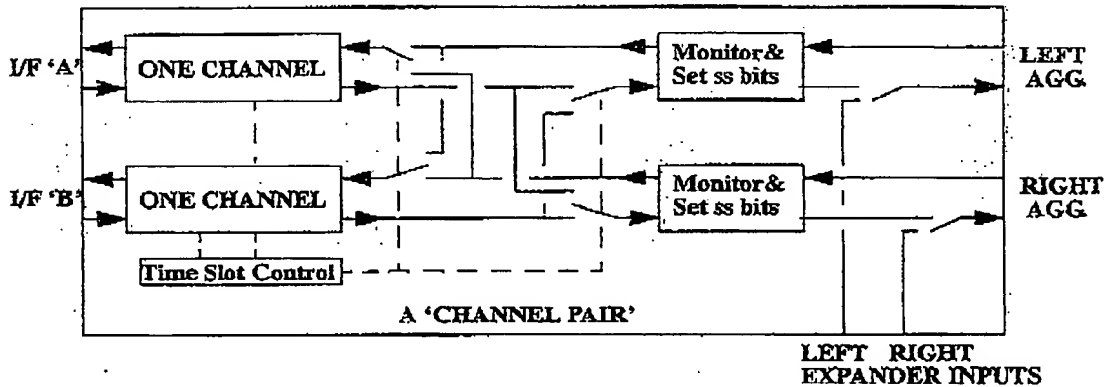
SATURN



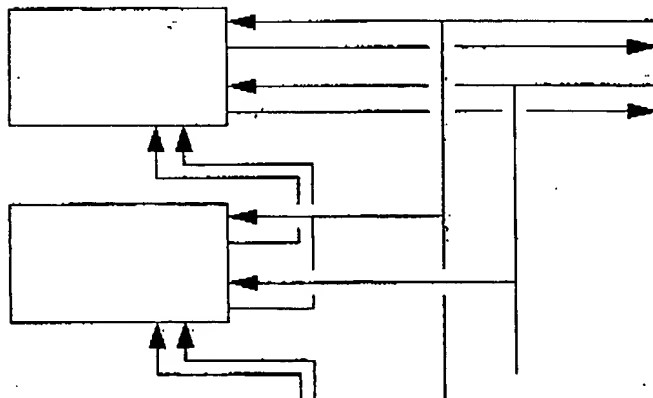
MULTIPLE CHANNEL PAIRS ON SATURN

FROM BPA TO "CHANNEL PAIR" BLOCK: FAN OUT ONLY

FROM "CHANNEL PAIR" TO BPA: PROPOSE "CHANNEL PAIR" INCORPORATES EXPANDER INPUTS AS:



- EXPANDER INPUTS RE-TIME 78M DATA
 - LEFT EXPANDER INPUT IS SELECTED EXCEPT FOR TIME SLOTS WHERE THE CHANNEL PAIR IS ACTIVE ON LEFT
 - RIGHT EXPANDER INPUT IS ACTIVE EXCEPT FOR TIME SLOTS WHERE THE CHANNEL PAIR IS ACTIVE ON RIGHT
- CONNECT TOGETHER AS:



NOTE: SUGGEST ONE 'STOLEN' A2 BYTE (SECTION INTEGRITY) PER CHANNEL.

IDEALLY WANT ONLY ONE CONNECTION ID CHECK (POLLED) PER CARD

Appendix: A: Virtual Concatenation

A1: Definition

The Sonet / SDH payload carried over Telecomms equipment is divided into a number of channels each represented by a virtual container (VC). Each of these has a defined bandwidth, so in order to carry a signal which has a bandwidth greater than that of a VC, the user has two choices. Either go to the next size up of VC, or use multiple Vcs to carry the signal. The standard for SDH envisaged that in the range VC-12 (approx 2 Mbit/s) up to VC-4 (approx 150 Mbit/s), a user would choose the next size of container. For rates above VC-4, the standard provides 'real concatenated' bandwidth, which provides channels with bandwidth of multiples of VC-4. However, neither of these measures meets user needs. This is because for the lower rates, the steps in container size are too large. For instance the next size up from VC-12 at approx. 2 Mbit/s is VC-2 at approx 8 Mbit/s, followed by VC-3 at approx. 50 Mbit/s. VC-2 is not commonly supported by equipment in the network, leaving a user who wishes to transport a 4 Mbit/s signal having to use (and pay for) a VC-3 with enormous wastage. For rates above VC-4, although the standards support VC-4-nc (any multiple of VC-4), the reality is that most equipment deployed in the network does not support this.

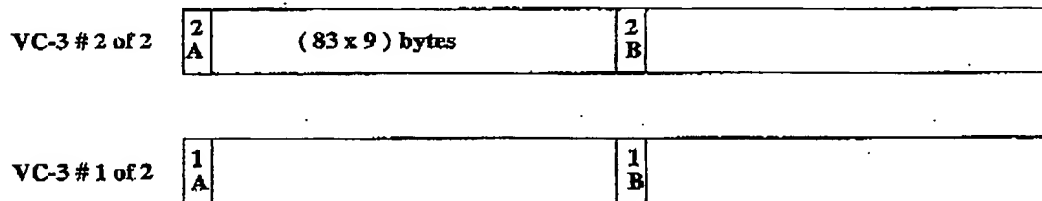
A user needing to efficiently transport signals having a bandwidth which either fits between the available VC bandwidths or exceeds VC-4 bandwidth, has to use a multiple of VCs to carry the signal. Because each VC is accessed and traverses the network independantly, this is called 'virtual' concatenation. In order to use virtual concatenation, the user has to take account of the fact that the network is unaware that this is happening. The different VCs that are used, may be routed differently by the network, and even if routed together will experience different delays.

A2: Using Virtual Concatenation

A.2.1: Virtual Concatenation: Originating End

To describe the principle of virtual concatenation, the example of concatenating two VC-3s will be used. Firstly at the originating end, two VC-3s are created to carry the payload. They will be created with timing to suit the local equipment multi-frame sync and probably with a convenient pointer value. Both will be created 'simultaneously'.

Only the payload region is illustrated.



Virtual concatenation overhead. Stream no (1-2) and Sequence no (A-B) shown.

One column of the payload at a known location is used to append Virtual Concatenation overhead. This includes a sequence number which identifies the SDH (125 μ s) frame. The number of frames this number must increment over before repeating is determined by the differential

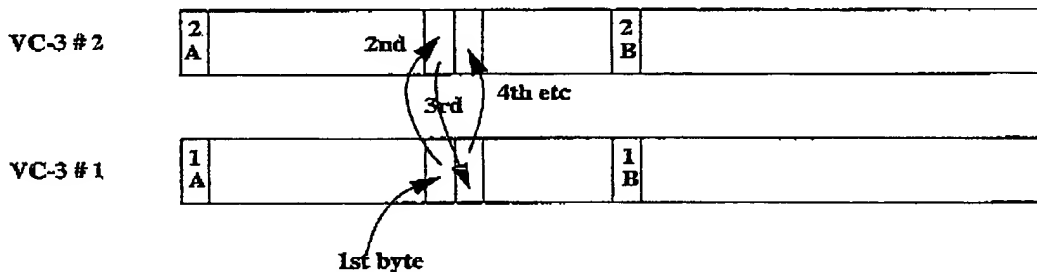
delay the two VC-3s may incur crossing a network. Nominally if the differential delay is N frames, then the numbers must run over $2N$ frames before repeating. In practice some allowance must be made for the payload bytes not being uniformly spread over the 125 μ S interval and the number may have to run over one more frame than nominal.

Differential delay can have two causes. For two VC-3s which run over the same physical route (fibre), the differential delay may be caused by intervening NEs. This will be null for NEs which manipulate only VC-4s (if the two VC-3s are in the same VC-4), otherwise it will be due to variations in fill of LO ptr processor buffers. (Any time switching should only add 1 byte delay ~ unless its a very badly designed time switch). For two VC-3s which run over different routes (which could happen if a path protection switch only occurs on one VC), then the differential delay is due to different physical distances (fibre delay) plus the absolute delays of intervening NEs.

[Note:

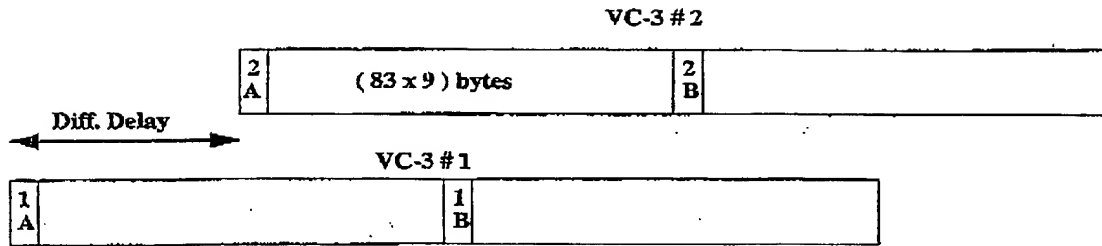
A data packet to be carried in the concatenated payload can start at any time relative to the VC. It is not desirable to incur delays and storage in holding the data frame to the next VC start. It is inserted into the payload as shown (but note that encoding of data frames is covered in a separate Appendix).

]

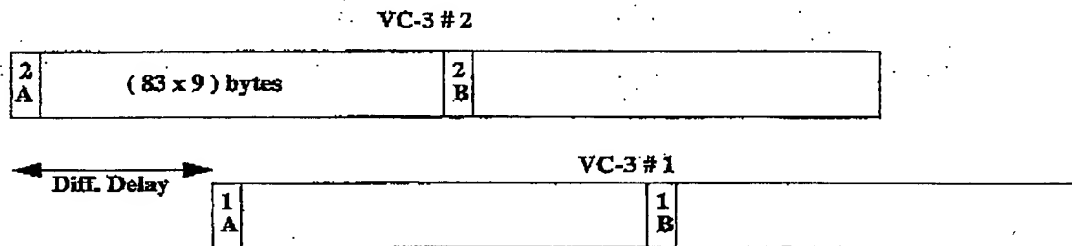


A.2.2: Virtual Concatenation : Destination End

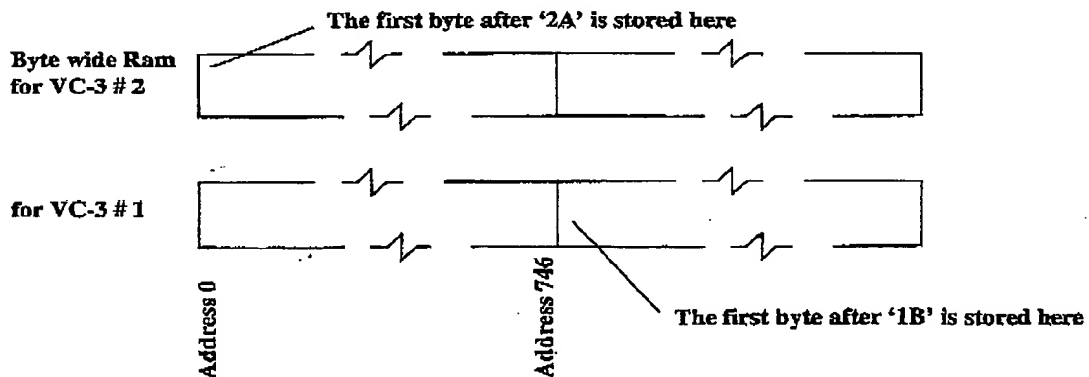
When the two VC-3s arrive at the destination equipment, they will have (in the case of Saturn) been pointer processed. Their pointers will therefore be at known locations, but the VCs will have suffered different delays. This is illustrated below, assuming that the delay is less than 125 μ S (for ease of illustration).



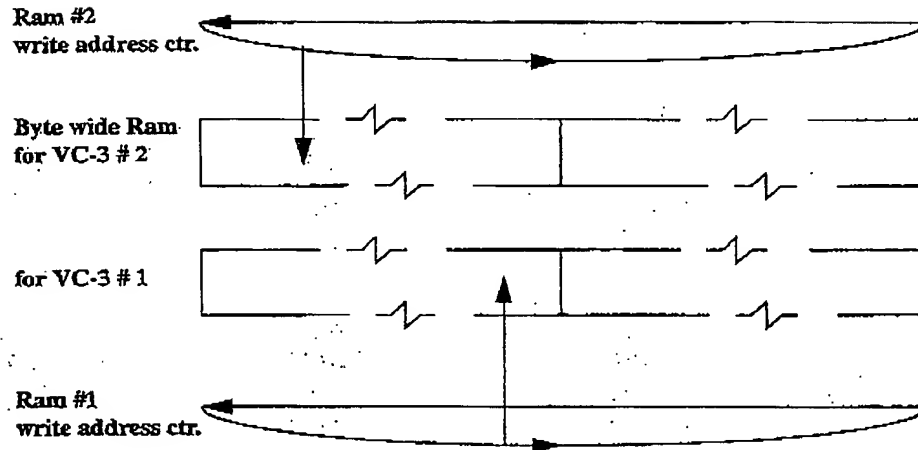
They could arrive as shown above,
or as shown below.



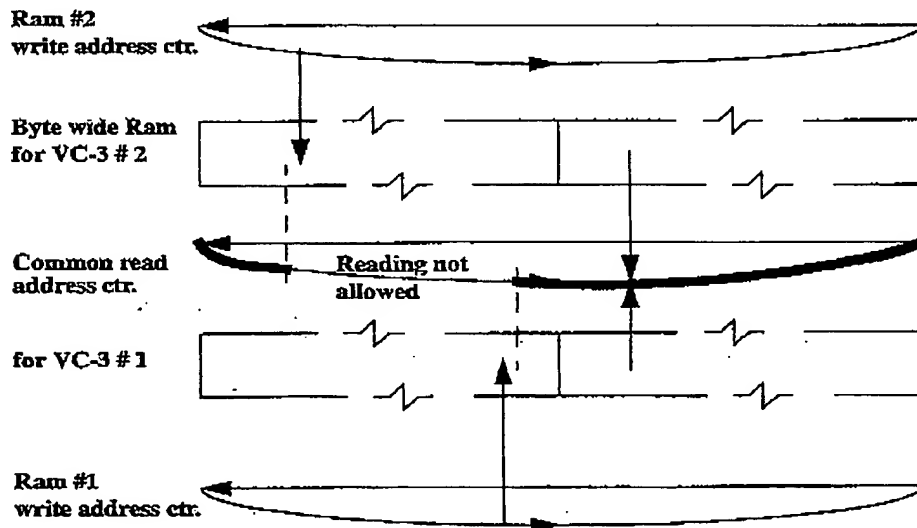
A pointer receiver is necessary to locate the payload, the payload is then identified by the sequence number as 'A' or 'B' and it is written to a ram. The ram is organised as follows:



Assume VC-3 #2 arrives after VC-3 #1. At an instant in time the locations illustrated below will be being written. In order to recover the original data, it must be read in its original alignment from locations where data from the same frame of both VCs is simultaneously valid.



The tricky bit is to control a read counter (this is rather like a pointer generator), so that simultaneous reading of both Rams occurs in a legal region illustrated below:



On the face of it, to control the read address counter, one waits for the last VC of the two to arrive (when it writes to address 0). The read address counter can then trickle along behind the write address of #2 above, so long as it doesn't fall so far behind that the write counter of #1 catches up with it. The problem is that if the system started up, just after #1 had written to address 0, it would think #2 was the first arriving frame and #1 arrived later. This is why the sequence number has to run for > twice the designed differential delay. If 'last' arrives more than the designed value after 'first', the read address counter needs resetting. Controlling the

read address counter in normal operation should be hitless, so it must be controlled by incrementing faster or slower, not by resetting to a new value. This can be achieved by clock gap-ping.

Note: it may not be necessary for the payload ram to be twice as large as the differential delay, but it's a lot easier to think about if it is.

Note: if the virtually concatenated VCs suffer a differential delay larger than the range of the sequence numbering, this can only be detected by corruption of the payload data.

Note: This is an example to illustrate the principle, the actual Virtual Concatenation overhead is described later.

Note: Virtual Concatenation overhead is not read, believed and acted on separately for each frame. The stream identification (VC-3 no. x of a total of y) is considered to indicate a fault if it is not the expected value for m1 successive frames. The sequence number is compared to a local sequence counter. Three unexpected sequence numbers in three consecutive frames is considered a fault, and causes the local sequence counter to hunt for a new alignment. Data extraction from the virtual concatenated containers is based on the local sequence counter.

Note: The choice of Virtual Concatenation overhead with its location determined by its place in the VC structure precludes the passage of concatenated payloads over PDH links.

A3: Virtual Concatenation Overhead

A.3.1: VC-3 and VC-4

One column located immediately after the path overhead column.

- 1st byte (after J1): Stream Number, 1-64
- 2nd byte (after B3): Total Number of Streams, 1-64
- 3rd byte (after C2): Sequence Number LSBs
- 4th byte (after G1): Sequence Number MSBs
- 5th thru 9th byte: Reserved

Note: Use of only the LSBs of the sequence number permits +/- 128 frames or +/- 16 mS differential delay.

A.3.2: VC-12

One byte immediately following V5 and one byte immediately following N2.

- Byte following V5: 4 bit Stream Number , 4 bit Total Number of Streams. Binary 0-15 used as Stream 1-16.
- Byte following N2: 8 bit Sequence Number.

Note: The 8 bit sequence number applies to 500 uS multiframes. This permits +/- 128 multiframes or +/- 64 mS differential delay.